明細書

再生装置、プログラム、再生方法

技術分野　本発明は、デジタル化された映画作品の再生と、アプリケーションの実行とを同時に実行する、再生制御技術の技術分野に属する発明であり、本再生制御技術を民生用の再生装置、プログラムに応用する場合の応用技術に深く係る。

背景技術

映画ビジネスには、デジタル化された映画作品と、ゲームアプリケーションとを１つのパッケージに収めて販売するというメディアミックスの形態がある。このゲームアプリケーションが、ゲームの登場人物をキャラクタにしており、デジタル化された映画作品の一部分と同時に実行されるのなら、映像再生と、ゲーム実行との双方による相乗効果で、映画作品の人気は一層高まる。

しかしながらアプリケーションには、程度の差はあれ、多かれ少なかれバグがある。バグが潜在していたため、装置がブラックアウトするのは、パソコンソフトの世界では”よくあること”で済まされるかもしれないが、民生機器の分野では品質問題になりかねない。品質問題への波及がありうるので、メーカ各社は、アプリケーションの実行と、デジタルストリーム再生とを同時に実行するような再生装置の実現に二の足を踏むことが多かった(注;ブラックアウトとは、装置のソフトウェアがフリーズして表示画面が真っ暗になる状態をいう)。

発明の開示

本発明の目的は、品質問題への波及を避けつつ、アプリケーションの実行と、デジタルストリーム再生とを同時に実行することができる再生装置を提供することである。

上記目的は、タイトルの再生と、アプリケーションの実行とを行う再生装置であって、タイトルに帰属するデジタルストリームを再生する再生制御エンジン部と、複数タイトル間の分岐を制御するモジュールマネージャと、1 つ以上

のアプリケーションを実行するモジュールとを備え、前記モジュールは、仮想マシン部と、アプリケーションマネージャを含み、前記アプリケーションマネージャは、1つ以上のアプリケーションが所定の状態になった際、タイトル実行が終了したとして解釈して終了処理を行い、前記モジュールマネージャは、前記タイトルの終了後、所定のタイトルを選択する、再生装置により達成される。

アプリケーションを含むが、デジタルストリームは含まないようなタイトルの終了時においても、所定のタイトルに分岐するという制御が可能になる。これによりアプリケーションプログラムがエラー終了したり、アプリケーションの起動に失敗したとしても、違和感がない制御を保証することができる。

こうした保証があるので、品質問題への波及に神経を尖らせているメーカ各社に対しても、ストリーム再生と、アプリケーション実行とを同時に実行する再生装置の開発を強く後押しすることができる。こうした強い後押しにより、再生装置の低価格化、多様化が進めば、BD-ROM コンテンツの充実化を図ることができるので、コンテンツ関連産業の発達を強く牽引することができる。

図面の簡単な説明

図1は、本発明に係る再生装置の使用行為についての形態を示す図である。

図2は、BD-ROM におけるファイル・ディレクトリ構成を示す図である。

図3は、AVClip 時間軸と、PL 時間軸との関係を示す図である。

図4は、4つの Clip_Information_file_name によりなされた一括指定を示す図である。

図5は、PLmark によるチャプター定義を示す図である。

図6は、SubPlayItem 時間軸上の再生区間定義と、同期指定とを示す図である。

図7（a）は、Movie オブジェクトの内部構成を示す図である。

図7（b）は、BD-J オブジェクトの内部構成を示す図である。

図7（c）は、Java アプリケーションの内部構成を示す図である。

図8（a）は、Java アーカイブファイルに収められているプログラム、デー

タを示す図である。

図８（ｂ）は、xlet プログラムの一例を示す図である。

図９（ａ）は、トップメニュー、title#1、title#2 といった一連のタイトルを示す図である。

図９（ｂ）は、PlayList#1、PlayList#2 の時間軸を足し合わせた時間軸を示す図である。

図１０は、本編タイトル、オンラインショッピングタイトル、ゲームタイトルという３つのタイトルを含むディスクコンテンツを示す図である。

図１１は、図１０に示した３つのタイトルの再生画像の一例を示す図である。

図１２（ａ）は、図１０の破線に示される帰属関係から各アプリケーションの生存区間をグラフ化した図である。

図１２（ｂ）は、図１２（ａ）の生存区間を規定するため、記述されたアプリケーション管理テーブルの一例を示す図である。

図１３（ａ）は、起動属性設定の一例を示す図である。

図１３（ｂ）は、他のアプリケーションからのアプリケーション呼出があって初めて起動するアプリケーション(application#2)を示す図である。

図１４（ａ）（ｂ）は、Suspend が有意義となるアプリケーション管理テーブル、生存区間の一例を示す図である。

図１５は、起動属性がとり得る三態様(Persistent、AutoRun、Suspend)と、直前タイトルにおけるアプリケーション状態の三態様(非起動、起動中、Suspend)とがとりうる組合せを示す図である。

図１６は、本発明に係る再生装置の内部構成を示す図である。

図１７（ａ）は、BD-ROM に存在している Java アーカイブファイルを、ローカルメモリ２９上でどのように識別するかを示す図である。

図１７（ｂ）は、図１７（ａ）の応用を示す図である。

図１８は、ROM２４に格納されたソフトウェアと、ハードウェアとからなる部分を、レイア構成に置き換えて描いた図である。

図１９は、Presentation Engine３１～モジュールマネージャ３４による処理を模式化した図である。

図２０は、アプリケーションマネージャ３６による処理を模式化した図である。

図２１は、ワークメモリ３７～Default Operation Manager４０を示す図である。

図２２は、アプリケーションマネージャ３６による分岐時の制御手順を示す図である。

図２３は、アプリケーション終了処理の処理手順を示すフローチャートである。

図２４は、アプリケーション終了の過程を模式的に示した図である。

図２５（ａ）は、PL 時間軸上に生存区間を定めたアプリケーション管理テーブルを示す図である。

図２５（ｂ）は、図２５（ａ）のアプリケーション管理テーブルに基づき、アプリケーションの生存区間を示した図である。

図２６（ａ）は、PL 時間軸から定まるタイトル時間軸を示す。

図２６（ｂ）は、メインとなるアプリケーションの生存区間から定まるタイトル時間軸を示す。

図２６（ｃ）は、複数アプリケーションの生存区間から定まるタイトル時間軸を示す図である。

図２７は、タイトル再生時におけるアプリケーションマネージャ３６の処理手順を示すフローチャートである。

図２８（ａ）は、BD-ROM により実現されるメニュー階層を示す図である。

図２８（ｂ）は、メニュー階層を実現するための MOVIE オブジェクトを示す図である。

図２９は、Index Table と、Index Table から各 Movie オブジェクトへの分岐とを模式化した図である。

図３０（ａ）は、図２９（ｂ）のように Index Table が記述された場合における分岐を示す。

図３０（ｂ）は、非 AV 系タイトルが強制終了した際における分岐を示す図である。

図３１は、モジュールマネージャ３４の処理手順を示すフローチャートである。

図３２は、アプリケーションマネージャ３６によるアプリケーション強制終了の動作例を示す図である。

図３３は、Playback Control Engine３２による PL 再生手順を示すフローチャートである。

図３４は、アングル切換、SkipBack,SkipNext の受付手順を示すフローチャートである。

図３５は、SkipBack,SkipNextAPI がコールされた際の処理手順を示すフローチャートである。

図３６は、Presentation Engine３１による処理手順の詳細を示すフローチャートである。

図３７は、SubPlayItem の再生手順を示すフローチャートである。

図３８は、第５実施形態に係るアプリケーションマネージャ３６の処理手順を示すフローチャートである。

図３９は、データ管理テーブルの一例を示す図である。

図４０は、BD-J オブジェクトが想定している実行モデルを示す図である。

図４１（ａ）は、ローカルメモリ２９における Java アーカイブファイル生存を示す生存区間を示す図である。

図４１（ｂ）は、図４１（ａ）での Java アーカイブファイル生存区間を規定するため、記述されたデータ管理テーブルを示す図である。

図４２は、カルーセル化による Java アーカイブファイル埋め込みを示す図である。

図４３（ａ）は、インターリーブ化による AVClip 埋め込みを示す図である。

図４３（ｂ）は、読込属性の３つの類型を示す図である。

図４４（ａ）は、データ管理テーブルの一例を示す図である。

図４４（ｂ）は、図４４（ａ）のデータ管理テーブルの割り当てによるローカルメモリ２９の格納内容の変遷を示す図である。

図４５（ａ）は、新旧再生装置におけるローカルメモリ２９のメモリ規模を

対比して示す図である。

図４５（ｂ）は、読込優先度が設定されたデータ管理テーブルの一例を示す図である。

図４６は、アプリケーションマネージャ３６によるプリロード制御の処理手順を示す図である。

図４７（ａ）は、applicationID が同一であるが、読込優先度は互いに異なる複数のアプリケーションを規定するデータ管理テーブルの一例を示す図である。

図４７（ｂ）は、図４７（ａ）のデータ管理テーブルの割り当てによるローカルメモリ２９の格納内容の変遷を示す図である。

図４８（ａ）は、プリロードされるべきアプリケーション、ロードされるべきアプリケーションに同一の applicationID を付与するよう記述されたデータ管理テーブルの一例を示す図である。

図４８（ｂ）は、メモリ規模が小さい再生装置におけるローカルメモリ２９の格納内容の変遷を示す図である。

図４８（ｃ）は、メモリ規模が大きい再生装置におけるローカルメモリ２９の格納内容の変遷を示す図である。

図４９は、データ管理テーブルに基づくアプリケーションマネージャ３６によるロード処理の処理手順を示す図である。

図５０は、アプリケーション q の生存区間に、現在の再生時点が到達した場合のアプリケーションマネージャ３６による処理手順を示す図である。

図５１は、Java 仮想マシン３８によるアプリケーションの読み込みがどのようにして行われるかを模式化した図である。

図５２（ａ）は、第７実施形態に係る BD-J オブジェクトの内部構成を示す図である。

図５２（ｂ）は、プレイリスト管理テーブルの一例を示す図である。

図５２（ｃ）は、分岐先タイトルのプレイリスト管理テーブルにおいて、再生属性が AutoPlay に設定された PL が存在する場合、再生装置がどのような処理を行うかを示す図である。

図５３（ａ）は、再生属性が非自動再生を示すよう設定された場合の非 AV 系タイトルにおけるタイトル時間軸を示す図である。

図５３（ｂ）は、再生属性が AutoPlay に設定された非 AV 系タイトルのタイトル時間軸を示す図である。

図５３（ｃ）は、プレイリスト管理テーブルにおいて再生属性が" AutoPlay"を示すよう設定され、アプリケーションが強制終了した場合を示す図である。

図５３（ｄ）は、プレイリスト管理テーブルにおいて再生属性が" AutoPlay"を示すよう設定され、メインアプリの起動に失敗したケースを示す図である。

図５４は、第７実施形態に係るアプリケーションマネージャ３６の処理手順を示すフローチャートである。

図５５は、プレイリスト管理テーブルにおいて" 再生属性＝AutoPlay" に設定されることにより、どのような再生が行われるかを模式化した図である。

図５６（ａ）（ｂ）は、アプリケーションの扱いと、起動属性との関係を示した図である。

図５７は、第８実施形態に係る Java 仮想マシン３８によるアプリケーションの読み込みがどのようにして行われるかを模式化した図である。

図５８（ａ）（ｂ）は、第９実施形態に係る読込優先度の一例を示す図である。

図５９（ａ）は、グループ属性が付与されたデータ管理テーブルを示す図である。

図５９（ｂ）は、アプリケーション管理テーブルに基づくローカルメモリ２９に対するアクセスを示す図である。

図６０は、アプリケーション管理テーブルの割当単位のバリエーションを示す図である。

発明を実施するための最良の形態

(第１実施形態)

以降、本発明に係る再生装置の実施形態について説明する。先ず始めに、本発明に係る再生装置の実施行為のうち、使用行為についての形態を説明する。図１は、本発明に係る再生装置の、使用行為についての形態を示す図である。

図１において、本発明に係る再生装置は再生装置２００であり、テレビ３００、リモコン４００と共にホームシアターシステムを形成する。

この BD-ROM１００は、再生装置２００、リモコン３００、テレビ４００により形成されるホームシアターシステムに、映画作品を供給するという用途に供される。

以上が本発明に係る再生装置の使用形態についての説明である。

続いて本発明に係る再生装置の再生の対象となる、記録媒体である BD-ROM について説明する。BD-ROM により、ホームシアターシステムに供給されるディスクコンテンツは、互いに分岐可能な複数タイトルから構成される。各タイトルは、1 つ以上のプレイリストと、このプレイリストを用いた動的な制御手順とからなる。

プレイリストとは、1 つ以上のデジタルストリームと、そのデジタルストリームにおける再生経路とから構成され、”時間軸”の概念をもつ BD-ROM 上のアクセス単位である。以上のプレイリストと、動的な制御手順とを包含しているため、タイトルはデジタルストリーム特有の時間軸の概念と、コンピュータプログラム的な性質とを併せもっている。

図２は、BD-ROM におけるファイル・ディレクトリ構成を示す図である。本図において BD-ROM には、Root ディレクトリの下に、BDMV ディレクトリがある。

BDMV ディレクトリには、拡張子 bdmv が付与されたファイル(index.bdmv,MovieObject.bdmv)と、拡張子 BD-J が付与されたファイル(00001.BD-J,00002.BD-J,00003.BD-J)がある。そしてこの BDMV ディレクトリの配下には、更に PLAYLIST ディレクトリ、CLIPINF ディレクトリ、STREAM ディレクトリ、BDAR ディレクトリと呼ばれる 4 つのサブディレクトリが存在する。

PLAYLIST ディレクトリには、拡張子 mpls が付与されたファイル(00001.mpls,00002.mpls,00003mpls)がある。

CLIPINF ディレクトリには、拡張子 clpi が付与されたファイル(00001.clpi,00002.clpi,00003.clpi)がある。

STREAMディレクトリには、拡張子m2tsが付与されたファイル(00001.m2ts,00002.m2ts,00003.m2ts)がある。

BDARディレクトリには、拡張子jarが付与されたファイル(00001.jar,00002.jar,00003jar)がある。以上のディレクトリ構造により、互いに異なる種別の複数ファイルが、BD-ROM上に配置されていることがわかる。

本図において拡張子.m2tsが付与されたファイル(00001.m2ts,00002.m2ts,00003.m2ts・・・・)は、AVClipを格納している。AVClipには、MainCLip、SubClipといった種別がある。MainCLipは、ビデオストリーム、オーディオストリーム、プレゼンテーショングラフィクスストリーム、インタラクティブグラフィクスストリームというような複数エレメンタリストリームを多重化することで得られたデジタルストリームである。

SubClipは、オーディオストリーム、グラフィクスストリーム、テキスト字幕ストリーム等、1つのエレメンタリストリームのみにあたるデジタルストリームである。

拡張子"clpi"が付与されたファイル(00001.clpi,00002.clpi,00003.clpi・・・・)は、AVClipのそれぞれに1対1に対応する管理情報である。管理情報故に、Clip情報は、AVClipにおけるストリームの符号化形式、フレームレート、ビットレート、解像度等の情報や、頭出し位置を示すEP_mapをもっている。

拡張子"mpls"が付与されたファイル(00001.mpls,00002.mpls,00003.mpls・・・・)は、プレイリスト情報を格納したファイルである。プレイリスト情報は、AVClipを参照してプレイリストを定義する情報である。プレイリストは、MainPath情報、PLMark情報、SubPath情報から構成される。

MainPath情報は、複数のPlayItem情報からなる。PlayItemとは、1つ以上のAVClip時間軸上において、In_Time,Out_Timeを指定することで定義される再生区間である。PlayItem情報を複数配置させることで、複数再生区間からなるプレイリスト(PL)が定義される。図3は、AVClipと、PLとの関係を示す図である。第1段目はAVClipがもつ時間軸を示し、第2段目は、PLがもつ時間軸を示す。PL情報は、PlayItem#1,#2,#3という3つのPlayItem情報

を含んでおり、これらPlayItem#1,#2,#3のIn_time,Out_timeにより、3つの再生区間が定義されることになる。これらの再生区間を配列させると、AVClip時間軸とは異なる時間軸が定義されることになる。これが第2段目に示すPL時間軸である。このように、PlayItem情報の定義により、AVClipとは異なる時間軸の定義が可能になる。

AVClipに対する指定は、原則1つであるが、複数AVClipに対する一括指定もあり得る。この一括指定は、PlayItem情報における複数のClip_Information_file_nameによりなされる。図4は、4つのClip_Information_file_nameによりなされた一括指定を示す図である。本図において第1段目～第4段目は、4つのAVClip時間軸(AVClip#1,#2,#3,#4の時間軸)を示し、第5段目は、PL時間軸を示す。PlayItem情報が有する、4つのClip_Information_file_nameにて、これら4つの時間軸が指定されている。こうすることで、PlayItemが有するIn_time,Out_timeにより、択一的に再生可能な4つの再生区間が定義されることになる。これにより、PL時間軸には、切り換え可能な複数アングル映像からなる区間(いわゆるマルチアングル区間)が定義されることになる。

PLmark情報は、PL時間軸のうち、任意の区間を、チャプターとして指定する情報である。図5は、PLmarkによるチャプター定義を示す図である。本図において第1段目は、AVClip時間軸を示し、第2段目はPL時間軸を示す。図中の矢印pk1,2は、PLmarkにおけるPlayItem指定(ref_to_PlayItem_Id)と、一時点の指定(mark_time_stamp)とを示す。これらの指定によりPL時間軸には、3つのチャプター(Chapter#1,#2,#3)が定義されることになる。

SubPath情報は、複数のSubPlayItem情報からなる。SubPlayItem情報は、SubClipの時間軸上にIn_Time,Out_Timeを指定することで再生区間を定義する。またSubPlayItem情報は、SubClip時間軸上の再生区間を、PL時間軸に同期させるという同期指定が可能であり、この同期指定により、PL時間軸と、SubPlayItem情報時間軸とは同期して進行することになる。図6は、SubPlayItem時間軸上の再生区間定義と、同期指定を示す図である。本図において第1段目は、PL時間軸を示し、第2段目はSubPlayItem時間軸を示す。

図中の SubPlayItem.IN_time は再生区間の始点を、SubPlayItem.Out_time は再生区間の終点をそれぞれ示す。これにより SubClip 時間軸上にも再生区間が定義されていることがわかる。矢印 Sn1 において Sync_PlayItem_Id は、PlayItem に対する同期指定を示し、矢印 Sn2 において sync_start_PTS_of_PlayItem は、PL 時間軸における PlayItem 上の一時点の指定を示す。

複数 AVClip の切り換えを可能とするマルチアングル区間や、AVClip－SubClip を同期させ得る同期区間の定義を可能とするのが、BD-ROM におけるプレイリスト情報の特徴である。以上の Clip 情報及びプレイリスト情報は、"静的シナリオ" に分類される。何故なら、以上の Clip 情報及びプレイリスト情報により、静的な再生単位である PL が定義されるからである。以上で静的シナリオについての説明を終わる。

続いて" 動的なシナリオ" について説明する。動的シナリオとは、AVClip の再生制御を動的に規定するシナリオデータである。" 動的に" というのは、再生装置における状態変化やユーザからのキーイベントにより再生制御の中身がかわることをいう。BD-ROM では、この再生制御の動作環境として 2 つのモードを想定している。1 つ目は、DVD 再生装置の動作環境と良く似た動作環境であり、コマンドベースの実行環境である。2 つ目は、Java 仮想マシンの動作環境である。これら 2 つの動作環境のうち 1 つ目は、HDMV モードと呼ばれる。2 つ目は、BD-J モードと呼ばれる。これら 2 つの動作環境があるため、動的シナリオはこのどちらかの動作環境を想定して記述される。HDMV モードを想定した動的シナリオは Movie オブジェクトと呼ばれ、管理情報により定義される。一方 BD-J モードを想定した動的シナリオは BD-J オブジェクトと呼ばれる。

先ず初めに Movie オブジェクトについて説明する。

＜Movie オブジェクト＞

Movie オブジェクトは、" タイトル" の構成要素であり、ファイル MovieObject.bdmv に格納される。図 7 （a）は、Movie オブジェクトの内部構成を示す図である。Movie オブジェクトは、属性情報、複数のナビゲーショ

ンコマンドからなるコマンド列からなる。

属性情報は、PL時間軸において、MenuCallがなされた際、MenuCall後の再生再開を意図しているか否かを示す情報(resume_intention_flag)、PL時間軸においてMenuCallをマスクするかを示す情報(menu_call_mask)、タイトルサーチをマスクするかを示す情報(title_search_flag)からなる。Movieオブジェクトは、”時間軸”+”プログラム的制御”という2つの性質を併せ持つことができるで、本編再生を実行するもの等、多様な種類のタイトルがこのMovieオブジェクトにより記述されることになる。

ナビゲーションコマンド列は、条件分岐、再生装置における状態レジスタの設定、状態レジスタの設定値取得等を実現するコマンド列からなる。Movieオブジェクトにおいて記述可能なコマンドを以下に示す。

PlayPLコマンド

書式:PlayPL(第1引数, 第2引数)

第1引数は、プレイリストの番号で、再生すべきPLを指定することができる。第2引数は、そのPLに含まれるPlayItemや、そのPLにおける任意の時刻、Chapter、Markを用いて再生開始位置を指定することができる。

PlayItemによりPL時間軸上の再生開始位置を指定したPlayPL関数をPlayPLatPlayItem()、

Chapterにより PL時間軸上の再生開始位置を指定したPlayPL関数をPlayPLatChapter()、

時刻情報により PL時間軸上の再生開始位置を指定したPlayPL関数をPlayPLatSpecified Time()という。

JMPコマンド

書式:JMP 引数

JMPコマンドは、現在の動的シナリオを途中で廃棄し(discard)、引数たる分岐先動的シナリオを実行するという分岐である。JMP命令の形式には、分岐先動的シナリオを直接指定している直接参照のものと、分岐先動的シナリオを間

接参照している間接参照のものがある。

Movie オブジェクトにおけるナビゲーションコマンドの記述は、DVD におけるナビゲーションコマンドの記述方式と良く似ているので、DVD 上のディスクコンテンツを、BD-ROM に移植するという作業を効率的に行うことができる。Movie オブジェクトについては、以下の国際公開公報に記載された先行技術が存在する。詳細については、本国際公開公報を参照されたい。

国際公開公報 W0 2004/074976

以上で Movie オブジェクトについての説明を終える。続いて BD-J オブジェクトについて説明する。

＜BD-J オブジェクト＞

拡張子 BD-J が付与されたファイル(00001.BD-J,00002.BD-J,00003.BD-J)は、BD-J オブジェクトを構成する。BD-J オブジェクトは、Java プログラミング環境で記述された、BD-J モードの動的シナリオである。図７（ｂ）は、BD-J オブジェクトの内部構成を示す図である。本図に示すように BD-J オブジェクトは、Movie オブジェクト同様の属性情報、アプリケーション管理テーブルからなる。属性情報を有している点で BD-J オブジェクトは Movie オブジェクトとほぼ同じである。Movie オブジェクトとの違いは、BD-J オブジェクトはコマンドが直接記述されていない点である。つまり Movie オブジェクトにおいて制御手順は、ナビゲーションコマンドにより直接記述されていた。これに対し BD-J オブジェクトでは、そのタイトルを生存区間としている Java アプリケーションをアプリケーション管理テーブル上に定めることにより、間接的に制御手順を規定している。このような間接的な規定により、複数タイトルにおいて制御手順を共通化するという、制御手順の共通化を効率的に行うことができる。

図７（ｃ）は、Java アプリケーションの内部構成を示す図である。本図にお

いてアプリケーションは、仮想マシンのヒープ領域(ワークメモリとも呼ばれる)にロードされた1つ以上のxletプログラムからなる。このワークメモリでは、1つ以上のスレッドが動作しており、ワークメモリにロードされたxletプログラム、及び、スレッドから、アプリケーションは構成されることになる。以上がアプリケーションの構成である。

このアプリケーションの実体にあたるのが、BDMVディレクトリ配下のBDARディレクトリに格納されたJavaアーカイブファイル(00001.jar,00002.jar)である。以降、Javaアーカイブファイルについて説明する。

Javaアーカイブファイル(00001.jar,00002.jar)は、Javaアプリケーションを構成するプログラム、データを格納したアーカイブファイルである。図8（a）は、アーカイブファイルにより収められているプログラム、データを示す図である。本図におけるデータは、枠内に示すディレクトリ構造が配置された複数ファイルを、javaアーカイバでまとめたものである。枠内に示すディレクトリ構造は、rootディレクトリ、javaディレクトリ、imageディレクトリとからなり、rootディにcommon.pkgが、javaディレクトリにaaa.class,bbb.classが、imageディレクトリに、menu.jpgが配置されている。javaアーカイブファイルは、これらをjavaアーカイバでまとめることで得られる。かかるデータは、BD-ROMからキャッシュに読み出されるにあたって展開され、キャッシュ上で、ディレクトリに配置された複数ファイルとして取り扱われる。Javaアーカイブファイルのファイル名における"xxxxx"という5桁の数値は、アプリケーションのID(applicationID)を示す。本Javaアーカイブファイルがキャッシュに読み出された際、このファイル名における数値を参照することにより、任意のJavaアプリケーションを構成するプログラム，データを取り出すことができる。

Javaアーカイブファイルにおいて1つにまとめられるファイルには、xletプログラムがある。

xletプログラムは、JMF(Java Media FrameWork)インターフェイスを利用することができるJavaプログラムである。xletプログラムは、キーイベント

を受信する EventListner 等、複数の関数からなり、JMF 等の方式に従って、受信したキーイベントに基づく処理を行う。

図８（ｂ）は、xlet プログラムの一例を示す図である。JMF A" BD://00001.mpls";は、PL を再生するプレーヤインスタンスの生成を Java 仮想マシンに命じるメソッドである。A.play は、JMF プレーヤインスタンスに再生を命じるメソッドである。かかる JMF プレーヤインスタンス生成は、JMF ライブラリに基づきなされる。xlet プログラムの記述は、BD-ROM の PL に限らず、時間軸をもったコンテンツ全般に適用可能な JMF の記述である。このような記述が可能であるので、Java プログラミングに長けたソフトハウスに、BD-J オブジェクト作成を促すことができる。

図８（ｂ）における JumpTItle();は、ファンクション API のコールである。このファンクション API は、他のタイトルへの分岐(図中では title#1)を再生装置に命じるものである。ここでファンクション API とは、BD-ROM 再生装置により供給される API(Appliation Interface)である。JumpTitle コマンドの他にも、ファンクション API のコールにより、BD-ROM 再生装置特有の処理を xlet プログラムに記述することができる。

BD-J モードにおいて PL 再生は、JMF インターフェイスにより規定される。この JMF プレーヤインスタンスは、PL 時間軸を定めるものだから、タイトル時間軸は、この JMF プレーヤインスタンスをもったタイトルから定まる。また

BD-J モードにおいてタイトルからタイトルへの分岐は JumpTitleAPI のコールにより規定される。JumpTitleAPI コールは、いわばタイトルの終了時点を定めるものなので、こうした JMF プレーヤインスタンス、JumpTitleAPI コールをもったアプリケーションが、BD-J モードにおいてタイトルの開始及び終了を律することになる。かかるアプリケーションを本編再生アプリケーションという。

以上が、BD-J モードにおける動的シナリオについての説明である。この BD-J モードにおける動的シナリオにより、PL 再生と、プログラム的制御とを併せもったタイトルが定義されることになる。尚、本実施形態においてアプリ

ケーションを構成するプログラム、データは、Java アーカイブファイルにまとめられたが、LZH ファイル、zip ファイルであってもよい。

＜タイトル時間軸＞

タイトルを構成する静的シナリオ、動的シナリオについて説明を終えたところで、これらによりどのような時間軸が定義されるかについて説明する。タイトルにより定義される時間軸は、"タイトル時間軸"と呼ばれる。タイトル時間軸とは、Movie オブジェクト、又は、BD-J オブジェクトにより再生が命じられる PL により構成される。ここで一例を挙げるのは、図９（ａ）のようなタイトルである。このタイトルは、トップメニュー→title#1→title#2→トップメニュー、トップメニュー→title#3→トップメニューという一連のタイトルである。かかるタイトルのうち、title#1 は PlayList#1、PlayList#2、title#2 が PlayList#3、title#3 が PlayList#4 の再生を命じるものなら、図９（ｂ）のように、PlayList#1、PlayList#2 の時間軸を足し合わせた時間軸を、title#1 はもつことになる。同様に title#2 は、PlayList#3 時間軸からなる時間軸を、title#3 は PlayList#4 時間軸からなる時間軸を持つことになる。これらタイトル時間軸における PL 時間軸ではシームレス再生が保証されるが、タイトル時間軸間ではシームレス再生の保証は必要でなくなる。Java アプリケーションを動作させるにあたっては、Java アプリケーションを、仮想マシンのワークメモリ上に存在させてもよい期間(サービス期間)を、こうしたタイトル時間軸上に定義せねばならない。BD-J モードにおいて Java アプリケーションを動作させるにあたっては、互いに分岐し合う時間軸上に、Java アプリケーションのサービス期間を定義せねばならない。このサービス期間の定義が、BD-ROM 向けのプログラミングを行うにあたっての留意点になる。

最後に、index.bdmv に格納された IndexTable について説明する。IndexTable は、タイトル番号と、Movie オブジェクト、BD-J オブジェクトとを対応づけるテーブルであり、動的シナリオから動的シナリオへの分岐の際、参照される間接参照用テーブルである。IndexTable は、複数ラベルのそれぞれに対する Index からなる。各 Index には、そのラベルに対応する動的シナリオの識別子が記述されている。こうした IndexTable を参照することで、Movie

オブジェクト、BD-J オブジェクトの違いを厳密に区別することなく、分岐を実現することができる。IndexTable については、以下の国際公開公報に詳細が記載されている。詳細については、本公報を参照されたい。

国際公開公報 WO 2004/025651 A1 公報

以上が BD-ROM に記録されているファイルについて説明である。

＜アプリケーション管理テーブル＞

JMF プレーヤインスタンス、JumpTitleAPI コールをもったアプリケーションが、タイトル時間軸を律することは上述した通りだが、JMF プレーヤインスタンスや JumpTitleAPI のコールをもたないその他のアプリケーションを、タイトル時間軸上で動作させる場合、時間軸の何処からアプリケーションによるサービスを開始し、時間軸の何処でアプリケーションによるサービスを終えるかという”サービスの開始点･終了点”を明確に規定することが重要になる。本実施形態では、アプリケーションによるサービスが開始してから、終了するまでを、”アプリケーションの生存”として定義する。アプリケーションの生存を定義するための情報は、BD-J オブジェクトにおけるアプリケーション管理テーブルに存在する。以降アプリケーション管理テーブルについてより詳しく説明する。

アプリケーション管理テーブル(AMT)は、各タイトルが有しているタイトル時間軸において、仮想マシンのワークメモリ上で生存し得るアプリケーションを示す情報である。ワークメモリにおける生存とは、そのアプリケーションを構成する xlet プログラムが、ワークメモリに読み出され、仮想マシンによる実行が可能になっている状態をいう。図７（ｂ）における破線矢印 at1 は、アプリケーション管理テーブルの内部構成をクローズアップして示す。この内部構成に示すように、アプリケーション管理テーブルは、『生存区間』と、そのタイトルを生存区間としているアプリケーションを示す『applicationID』と、そのアプリケーションの『起動属性』とからなる。

近い将来、実施されるであろうディスクコンテンツを題材に選んで、アプリ

ケーション管理テーブルにおける生存区間記述について、具体例を交えて説明する。ここで題材にするディスクコンテンツは、映像本編を構成する本編タイトル(title#1)、オンラインショッピングを構成するオンラインショッピングタイトル(title#2)、ゲームアプリケーションを構成するゲームタイトル(title#3)という、性格が異なる３つのタイトルを含むものである。図１０は、本編タイトル、オンラインショッピングタイトル、ゲームタイトルという３つのタイトルを含むディスクコンテンツを示す図である。本図における右側にはIndexTableを記述しており、左側には３つのタイトルを記述している。

右側における破線枠は、各アプリケーションがどのタイトルに属しているかという帰属関係を示す。３つのタイトルのうち title#1 は、application#1、application#2、application#3 という３つのアプリケーションからなる。title#2は、application#3、application#4 という２つのアプリケーション、title#3 は、application#5 を含む。図１１は、図１０に示した３つのタイトルの再生画像の一例を示す図である。これら３つのタイトルの再生画像において、図１１（ａ）（ｂ）の本編タイトル、オンラインショッピングタイトルには、ショッピングカートを模した映像(カート cr1)1 が存在するが、図１１（ｃ）のゲームタイトルには、カート映像が存在しない。カート cr1 は、本編タイトル、オンラインショッピングタイトルにおいて共通して表示しておく必要があるので、カートプログラムたる application#3 を、title#1、title#2 の双方で起動するようにしている。このように複数タイトルで起動するようなアプリケーションには、上述したカートアプリの他に、映画作品に登場するマスコットを模したエージェントアプリ、メニューコール操作に応じてメニュー表示を行うメニューアプリがある。

図１０の破線に示される帰属関係から各アプリケーションの生存区間をグラフ化すると、図１２（ａ）のようになる。本図において横軸は、タイトル時間軸であり、縦軸方向に各アプリケーションの生存区間を配置している。ここでapplication#1、application#2 は、title#1 のみに帰属しているので、これらの生存区間は、title#1 内に留まっている。application#4 は、title#2 のみに帰属しているので、これらの生存区間は、title#2 内に留まっている。application#5

は、title#3 のみに帰属しているので、これらの生存区間は、title#3 内に留まっている。application#3 は、title#1 及び title#2 に帰属しているので、これらの生存区間は、title#1－title#2 にわたる。この生存区間に基づき、アプリケーション管理テーブルを記述すると、title#1,#2,#2 のアプリケーション管理テーブルは図１２（ｂ）のようになる。このようにアプリケーション管理テーブルが記述されれば、title#1 の再生開始時において application#1、application#2、application#3 をワークメモリにロードしておく。そして title#2 の開始時に application#1、application#2 をワークメモリから削除して application#3 のみにするという制御を行う。これと同様に title#2 の再生開始時において application#4 をワークメモリにロードしておき、title#3 の開始時に application#3,#4 をワークメモリから削除するという制御を行いうる。

更に、title#3 の再生中において application#5 をワークメモリにロードしておき、title#3 の再生終了時に application#5 をワークメモリから削除するという制御を行いうる。

タイトル間分岐があった場合でも、分岐元－分岐先において生存しているアプリケーションはワークメモリ上に格納しておき、分岐元にはなく、分岐先にのみ存在するアプリケーションをワークメモリに読み込めば良いから、アプリケーションをワークメモリに読み込む回数は必要最低数になる。このように、読込回数を少なくすることにより、タイトルの境界を意識させないアプリケーション、つまりアンバウンダリなアプリケーションを実現することができる。

続いてアプリケーションの起動属性について説明する。起動属性には、自動的な起動を示す「AutoRun」、自動起動の対象ではないが、仮想マシンのワークメモリに置いて良いことを示す「Persistent」、仮想マシンのワークメモリにはおかれるが、CPU パワーの割り当ては不可となる「Suspend」がある。

「AutoRun」は、対応するタイトルの分岐と同時に、そのアプリケーションをワークメモリに読み込み、且つ実行する旨を示す生存区間である。あるタイトルから、別のタイトルへの分岐があると、アプリケーション管理を行う管理主体(アプリケーションマネージャ)は、その分岐先タイトルにおいて生存しており、かつ起動属性が AutoRun に設定されたアプリケーションを仮想マシン

のワークメモリに読み込み実行する。これによりそのアプリケーションは、タイトル分岐と共に自動的に起動されることになる。起動属性を AutoRun に設定しておくアプリケーションとしては、JMF プレーヤインスタンス及び JumpTitleAPI コールをもつようなアプリケーションが挙げられる。何故なら、このようなアプリケーションは、タイトル時間軸を律する側のアプリケーションであり、このようなアプリケーションを自動的に起動にしないと、タイトル時間軸の概念が曖昧になってしまうからである。

起動属性「Persistent」は、継続属性であり、分岐元 title におけるアプリケーションの状態を継続することを示す。またワークメモリにロードしてよいことを示す属性である。起動属性が「Persistent」である場合、この起動属性が付与されたアプリケーションは、他のアプリケーションからの呼び出しが許可されることになる。アプリケーション管理を行う管理主体(アプリケーションマネージャ)は、起動中のアプリケーションから呼出があると、そのアプリケーションの applicationID が、アプリケーション管理テーブルに記述されていて、起動属性が「Persistent」であるか否かを判定する。「Persistent」であれば、そのアプリケーションをワークメモリにロードする。一方、その呼出先アプリケーションの applicationID がアプリケーション管理テーブルに記述されていない場合、そのアプリケーションはワークメモリにロードされない。アプリケーションによる呼出は、この「Persistent」が付与されたアプリケーションに限られることになる。

「Persistent」は、起動属性を明示的に指定しない場合に付与されるデフォルトの起動属性であるから、あるアプリケーションの起動属性が無指定「－－」である場合、そのアプリケーションの起動属性の起動属性はこの Persistent であることを意味する。

これらの起動属性が、図１１のアプリケーションにおいてどのように記述されているかについて説明する。図１３は、図１２の３つのアプリケーションに対する起動属性の設定例である。図１２に示した３つのアプリケーションのうち application#2 は、図１３（ｂ）に示すように他のアプリケーションからのアプリケーション呼出があって初めて起動するアプリケーションであるとする。

残りの application#1、application#3 は、title#1 の開始と同時に自動的に起動されるアプリケーションであるとする。この場合、図１３（ａ）に示すように、アプリケーション管理テーブルにおける各アプリケーションの起動属性を、application#1、application#3 は「AutoRun」、application#2 は、「Persistent」と設定しておく。この場合、application#1、application#3 は、title#1 への分岐時において自動的にワークメモリにロードされ、実行されることになる。一方 application#2 は、起動属性が Persistent なので、「application#3 は仮想マシンのワークメモリ上にロードしてよいアプリケーション」であるとの消極的な意味に解される。故に、application#2 は、application#1 からの呼出があって初めて仮想マシンのワークメモリにロードされ、実行されることになる。以上の生存区間･起動属性により､仮想マシン上で動作し得るアプリケーションの数を 4 個以下に制限し、総スレッド数を 64 個以下に制限することが可能なので、アプリケーションの安定動作を保証することができる。

続いて Suspend について説明する。

Suspend とは、リソースは割り付けられているが、CPU パワーは割り当てられない状態にアプリケーションが置かれることをいう。かかる Suspend は、例えばゲームタイトルの実行中に、サイドパスを経由するという処理の実現に有意義である。図１４（ａ）（ｂ）は Suspend が有意義となる事例を示す図である。図１４（ｂ）に示すように、3 つのタイトル(title#1、title#2、title#3)があり、そのうち title#1、title#3 はゲームアプリを実行するが、途中の title#2 はサイドパスであり、映像再生を実現するものである。サイドパスでは、映像再生を実現する必要があるため、ゲームの実行を中断させることになる。ゲームアプリでは途中のスコア等が計数されているため、リソースの格納値は title#2 の前後で維持したい。この場合、title#2 の開始時点でゲームアプリを Suspend し、title#3 の開始時点で application#2 をレジュームするというようにアプリケーション管理テーブルを記述する。こうすることで title#2 において application#2 は、リソースは割り付けられているので、リソースの格納値は維持される。しかし、CPU パワーは割り当てられない状態なので仮想マシンにより application#2 は実行されることはない。これにより、ゲームタイトル

の実行中に、サイドパスを実行するという処理が実現される。

図１５は、起動属性がとり得る三態様(Persistent、AutoRun、Suspend)と、直前タイトルにおけるアプリケーション状態の三態様(非起動、起動中、Suspend)とがとりうる組合せを示す図である。直前状態が”非起動”である場合、起動属性が”AutoRun”であるなら、分岐先タイトルにおいてそのアプリケーションは、起動されることになる。

直前状態が”非起動”であり、起動属性が”Persistent”、”Suspend”であるなら、分岐先タイトルにおいてそのアプリケーションは、何ももせず、状態を継続することになる。

直前状態が”起動中”である場合、起動属性が”Persistent”、”AutoRun”であるなら、分岐先タイトルにおいてそのアプリケーションは、何もせず、状態を継続することになる。

起動属性が”Suspend”であるなら、アプリケーションの状態は Suspend されることになる。直前状態が”Suspend”である場合、分岐先タイトルの起動属性が”Suspend”なら Suspend を維持することになる。”Persistent”、”AutoRun”であるなら、分岐先タイトルにおいてそのアプリケーションは、レジュームすることになる。アプリケーション管理テーブルにおいて生存区間及び起動属性を定義することにより、タイトル時間軸の進行に沿って、Java アプリケーションを動作させるという同期制御が可能になり、映像再生と、プログラム実行とを伴った、様々なアプリケーションを世に送り出すことができる。以上が記録媒体についての説明である。続いて本発明に係る再生装置について説明する。

図１６は、本発明に係る再生装置の内部構成を示す図である。本発明に係る再生装置は、本図に示す内部に基づき、工業的に生産される。本発明に係る再生装置は、主としてシステム LSI と、ドライブ装置という２つのパーツからなり、これらのパーツを装置のキャビネット及び基板に実装することで工業的に生産することができる。システム LSI は、再生装置の機能を果たす様々な処理部を集積した集積回路である。こうして生産される再生装置は、BD-ROM ドライブ１、リードバッファ２、デマルチプレクサ３、ビデオデコーダ４、ビデ

オプレーン５、P-Graphics デコーダ９、Presentation Graphics プレーン１０、合成部１１、フォントゼネレータ１２、I-Graphics デコーダ１３、スイッチ１４、Interactive Graphics プレーン１５、合成部１６、HDD１７、リードバッファ１８、デマルチプレクサ１９、オーディオデコーダ２０、シナリオメモリ２１、CPU２２、キーイベント処理部２３、命令 ROM２４、スイッチ２５、CLUT 部２６、CLUT 部２７、PSR セット２８、ローカルメモリ２９から構成される。

BD-ROM ドライブ１は、BD-ROM のローディング／イジェクトを行い、BD-ROM に対するアクセスを実行する。

リードバッファ２は、FIFO メモリであり、BD-ROM から読み出された TS パケットが先入れ先出し式に格納される。

デマルチプレクサ(De-MUX)３は、リードバッファ２から TS パケットを取り出して、この TS パケットを構成する TS パケットを PES パケットに変換する。そして変換により得られた PES パケットのうち、CPU２２から設定された PID をもつものをビデオデコーダ４、オーディオデコーダ２０、P-Graphics デコーダ９、I-Graphics デコーダ１３のどれかに出力する。

ビデオデコーダ４は、デマルチプレクサ３から出力された複数 PES パケットを復号して非圧縮形式のピクチャを得てビデオプレーン５に書き込む。

ビデオプレーン５は、非圧縮形式のピクチャを格納しておくためのプレーンである。プレーンとは、再生装置において一画面分の画素データを格納しておくためのメモリ領域である。再生装置に複数のプレーンを設けておき、これらプレーンの格納内容を画素毎に加算して、映像出力を行えば、複数の映像内容を合成させた上で映像出力を行うことができる。ビデオプレーン５における解像度は 1920×1080 であり、このビデオプレーン５に格納されたピクチャデータは、16 ビットの YUV 値で表現された画素データにより構成される。

P-Graphics デコーダ９は、BD-ROM、HDD１７から読み出されたプレゼンテーショングラフィクスストリームをデコードして、非圧縮グラフィクスを Presentation Graphics プレーン１０に書き込む。グラフィクスストリームのデコードにより、字幕が画面上に現れることになる。

Presentation Graphics プレーン１０は、一画面分の領域をもったメモリであり、一画面分の非圧縮グラフィクスを格納することができる。本プレーンにおける解像度は 1920×1080 であり、Presentation Graphics プレーン１０中の非圧縮グラフィクスの各画素は８ビットのインデックスカラーで表現される。CLUT(Color Lookup Table)を用いてかかるインデックスカラーを変換することにより、Presentation Graphics プレーン１０に格納された非圧縮グラフィクスは、表示に供される。

合成部１１は、非圧縮状態のピクチャデータ(i)を、Presentation Graphics プレーン１０の格納内容と合成する。

フォントゼネレータ１２は、文字フォントを用いて textST ストリームに含まれるテキストコードをビットマップに展開する。

I-Graphics デコーダ１３は、BD-ROM 又は HDD１７から読み出されたインタラクティブグラフィクスストリームをデコードして、非圧縮グラフィクスを Interactive Graphics プレーン１５に書き込む。

スイッチ１４は、フォントゼネレータ１２が生成したフォント列、P-Graphics デコーダ９のデコードにより得られたグラフィクスの何れかを選択的に Presentation Graphics プレーン１０に書き込むスイッチである。

Interactive Graphics プレーン１５は、I-Graphics デコーダ１３によるデコードで得られた非圧縮グラフィクスが書き込まれる。

合成部１６は、Interactive Graphics プレーン１０の格納内容と、合成部８の出力である合成画像(非圧縮状態のピクチャデータと、Presentation Graphics プレーン７の格納内容とを合成したもの)とを合成する。

HDD１７は、ネットワーク等を介してダウンロードされた SubClip、Clip 情報、プレイリスト情報が格納される内蔵媒体である。この HDD１７中のプレイリスト情報は BD-ROM 及び HDD１７のどちらに存在する Clip 情報であっても、指定できる点で異なる。この指定にあたって、HDD１７上のプレイリスト情報は、BD-ROM 上のファイルをフルパスで指定する必要はない。本 HDD１７は、BD-ROM と一体になって、仮想的な１つのドライブ(バーチャルパッケージと呼ばれる)として、再生装置により認識されるからである。故に、

PlayItem 情報における Clip_Information_file_name 及び SubPlayItem 情報の Clip_Information_file_name は、Clip 情報の格納したファイルのファイルボディにあたる5桁の数値を指定することにより、HDD１７、BD-ROM 上の AVClip を指定することができる。この HDD の記録内容を読み出し、BD-ROM の記録内容と動的に組み合わせることにより、様々な再生のバリエーションを産み出すことができる。

リードバッファ１８は、FIFO メモリであり、HDD１７から読み出された TS パケットが先入れ先出し式に格納される。

デマルチプレクサ(De-MUX)１９は、リードバッファ１８から TS パケットを取り出して、TS パケットを PES パケットに変換する。そして変換により得られた PES パケットのうち、所望の streamPID をもつものをフォントゼネレータ１２に出力する。

オーディオデコーダ２０は、デマルチプレクサ１９から出力された PES パケットを復号して、非圧縮形式のオーディオデータを出力する。

シナリオメモリ２１は、カレントの PL 情報やカレントの Clip 情報を格納しておくためのメモリである。カレント PL 情報とは、BD-ROM に記録されている複数 PL 情報のうち、現在処理対象になっているものをいう。カレント Clip 情報とは、BD-ROM に記録されている複数 Clip 情報のうち、現在処理対象になっているものをいう。

CPU２２は、命令 ROM２４に格納されているソフトウェアを実行して、再生装置全体の制御を実行する。

キーイベント処理部２３は、リモコンや再生装置のフロントパネルに対するキー操作に応じて、その操作を行うキーイベントを出力する。

命令 ROM２４は、再生装置の制御を規定するソフトウェアを記憶している。

スイッチ２５は、BD-ROM 及び HDD１７から読み出された各種データを、リードバッファ２、リードバッファ１８、シナリオメモリ２１、ローカルメモリ２９のどれかに選択的に投入するスイッチである。

CLUT 部２６は、ビデオプレーン５に格納された非圧縮グラフィクスにおけるインデックスカラーを、Y,Cr,Cb 値に変換する。

CLUT部２７は、Interactive Graphicsプレーン１５に格納された非圧縮グラフィクスにおけるインデックスカラーを、Y,Cr,Cb値に変換する。

PSRセット２８は、再生装置に内蔵されるレジスタであり、64個のPlayer Status Register(PSR)と、4096個のGeneral Purpose Register（GPR)とからなる。Player Status Registerの設定値(PSR)のうち、PSR4～PSR8は、現在の再生時点を表現するのに用いられる。

PSR4は、1～100の値に設定されることで、現在の再生時点が属するタイトルを示し、0に設定されることで、現在の再生時点がトップメニューであることを示す。

PSR5は、1～999の値に設定されることで、現在の再生時点が属するチャプター番号を示し、0xFFFFに設定されることで、再生装置においてチャプター番号が無効であることを示す。

PSR6は、0～999の値に設定されることで、現在の再生時点が属するPL(カレントPL)の番号を示す。

PSR7は、0～255の値に設定されることで、現在の再生時点が属するPlay Item(カレントPlay Item)の番号を示す。

PSR8は、0～0xFFFFFFFFの値に設定されることで、45KHzの時間精度を用いて現在の再生時点(カレントPTM(Presentation TiMe))を示す。以上のPSR4～PSR8により、現在の再生時点が特定されることになる。

ローカルメモリ２９は、BD-ROMからの読み出しは低速である故、BD-ROMの記録内容を一時的に格納しておくためのキャッシュメモリである。かかるローカルメモリ２９が存在することにより、BD-Jモードにおけるアプリケーション実行は、効率化されることになる。図１７（ａ）は、BD-ROMに存在しているJavaアーカイブファイルを、ローカルメモリ２９上でどのように識別するかを示す図である。図１７（ａ）の表は、左欄にBD-ROM上のファイル名を、右欄にローカルメモリ２９上のファイル名をそれぞれ示している。これら右欄、左欄を比較すれば、ローカルメモリ２９におけるファイルは、ディレクトリ指定"BDJA"を省いたファイルパスで指定されていることがわかる。

図１７（ｂ）は、図１７（ａ）の応用を示す図である。本応用例は、ヘッダ

＋データという形式で、ファイルに格納されているデータを格納するというものである。何をヘッダに用いるかというと、ローカルメモリ２９におけるファイルパスを用いる。図１７（ｂ）に示したように、ローカルメモリ２９ではBD-ROM におけるファイルパスの一部を省略したものをファイルパスに用いるから、当該ファイルパスをヘッダに格納することで、各データにおけるBD-ROM 上の所在を明らかにすることができる。

以上が、本実施形態に係る再生装置のハードウェア構成である。続いて本実施形態に係る再生装置のソフトウェア構成について説明する。

図１８は、ROM２４に格納されたソフトウェアと、バードウェアとからなる部分を、レイア構成に置き換えて描いた図である。本図に示すように、再生装置のレイア構成は、以下の a),b),c),d-1),d-2),e),f)からなる。つまり、

a)物理的なハードウェア階層の上に、

b)AVClip による再生を制御する Presentation Engine３１、

c)プレイリスト情報及び Clip 情報に基づく再生制御を行う Playback Control Engine３２、

という２つの階層があり、

最上位の階層に

e)タイトル間の分岐を実行するモジュールマネージャ３４がある。

これら HDMV モジュール３３、モジュールマネージャ３４の間に、

d-1)Movie オブジェクトの解読･実行主体である HDMV モジュール３３と、

d-2)BD-J オブジェクトの解読･実行を行う BD-J モジュール３５とが同じ階層に置かれている。

BD-J モジュール３５は、いわゆる Java プラットフォームであり、ワークメモリ３７を含む Java 仮想マシン３８を中核にした構成になっていて、アプリケーションマネージャ３６、Event Listner Manager３９、Default Operation Manager４０から構成される。先ず初めに、Presentation Engine３１～モジュールマネージャ３４について説明する。図１９は、Presentation Engine３１～モジュールマネージャ３４による処理を模式化した図である。

Presentation Engine３１は、AV 再生機能を実行する。再生装置の AV 再生

機能とは、DVD プレーヤ、CD プレーヤから踏襲した伝統的な機能群であり、再生開始(Play)、再生停止(Stop)、一時停止(Pause On)、一時停止の解除(Pause Off)、Still 機能の解除(still off)、速度指定付きの早送り(Forward Play(speed))、速度指定付きの巻戻し(Backward Play(speed))、音声切り換え(Audio Change)、副映像切り換え(Subtitle Change)、アングル切り換え(Angle Change)といった機能である。AV 再生機能を実現するべく、Presentation Engine３１は、リードバッファ２上に読み出された AVClip のうち、所望に時刻にあたる部分のデコードを行うよう、ビデオデコーダ４、P-Graphics デコーダ９、I-Graphics デコーダ１３、オーディオデコーダ２０を制御する。所望の時刻として PSR8(カレント PTM)に示される箇所のデコードを行わせることにより、AVClip において、任意の時点を再生を可能することができる。図中の◎1 は、Presentation Engine３１によるデコード開始を模式化して示す。

再生制御エンジン(Playback Control Engine(PCE))３２は、プレイリストの再生機能(i)、再生装置における状態取得／設定機能(ii)といった諸機能を実行する。PL の再生機能とは、Presentation Engine３１が行う AV 再生機能のうち、再生開始や再生停止を、カレント PL 情報及び Clip 情報に従って行わせることをいう。これら機能(i)～(ii)は、HDMV モジュール３３～BD-J モジュール３５からのファンクションコールに応じて実行する。つまり再生制御エンジン３２は、ユーザ操作による指示、レイヤモデルにおける上位層からの指示に応じて、自身の機能を実行する。図１９において、◎2,◎3 が付された矢印は、Clip 情報及びプレイリスト情報に対する Playback Control Engine３２の参照を模式的に示す。

HDMV モジュール３３は、MOVIE モードの実行主体であり、モジュールマネージャ３４から分岐先を構成する Movie オブジェクトが通知されれば、分岐先タイトルを構成する Movie オブジェクトをローカルメモリ２９に読み出して、この Movie オブジェクトに記述されたナビゲーションコマンドを解読し、解読結果に基づき Playback Control Engine３２に対するファンクションコールを実行する。図１９において▽2,▽3,▽4 が付された矢印は、モジュールマネージャ３４からの分岐先 Movie オブジェクトの通知(2)、Movie オブジェクト

に記述されたナビゲーションコマンドの解読(3)、Playback Control Engine３２に対するファンクションコール(4)を、模式的に示している。

モジュールマネージャ３４は、BD-ROMから読み出されたIndex Tableを保持して、分岐制御を行う。この分岐制御は、JumpTitleコマンドをHDMVモジュール３３が実行した場合、又は、タイトルジャンプAPIがBD-Jモジュール３５からコールされた場合、そのジャンプ先となるタイトル番号を受け取って、そのタイトルを構成するMovieオブジェクト又はBD-Jオブジェクトを HDMVモジュール３３又はBD-Jモジュール３５に通知するというものである。図中の▽0,▽1,▽2が付された矢印は、JumpTitleコマンドの実行(0)、モジュールマネージャ３４によるIndexTable参照(1)、分岐先Movieオブジェクト(2)の通知を模式的に示している。

以上がPresentation Engine３１～モジュールマネージャ３４についての説明である。続いてアプリケーションマネージャ３６について、図２０を参照しながら説明する。図２０は、アプリケーションマネージャ３６を示す図である。

アプリケーションマネージャ３６は、アプリケーション管理テーブルを参照したアプリケーションの起動制御、タイトルの正常終了時における制御を実行する。

起動制御とは、モジュールマネージャ３４から分岐先となるBD-Jオブジェクトが通知される度に、そのBD-Jオブジェクトを読み出し、そのBD-Jオブジェクト内のアプリケーション管理テーブルを参照してローカルメモリ２９アクセスを行う。そして現在の再生時点を生存区間とするアプリケーションを構成するxletプログラムを、ワークメモリに読み出すという制御である。図２０における☆1,☆2,☆3は、起動制御における分岐先BD-Jオブジェクトの通知(1)、アプリケーション管理テーブル参照(2)、Java 仮想マシン３８に対する起動指示を模式化して示す。この起動指示によりJava 仮想マシン３８は、ローカルメモリ２９からワークメモリ３７にxletプログラムを読み出す(☆5)。

タイトルの終了制御には、正常終了時の制御と、異常終了時の制御とがある。正常終了時の制御には、タイトルを構成するアプリケーションによりジャンプタイトルAPIがコールされて、分岐先タイトルへの切り換えを分岐制御の主体

(モジュールマネージャ３４)に要求するという制御がある。この終了制御における、モジュールマネージャ３４通知を模式化して示したのが矢印☆6である。ここでタイトルを正常終了するにあたって、タイトルを構成するアプリケーションは、起動されたままであってもよい。何故なら、アプリケーションを終了するか否かは、分岐先タイトルにおいて判断するからである。本実施形態では深く触れないが、アプリケーションマネージャ３６は、BD-ROM からローカルメモリ２９に Java アーカイブファイルを読み出す(8)との処理を行う。このローカルメモリ２９への読み出しを模式化したのが☆8である。

以上がアプリケーションマネージャ３６についての説明である。続いてワークメモリ３７～Default Operation Manager４０について、図２１を参照しながら説明する。

ワークメモリ３７は、アプリケーションを構成する xlet プログラムが配置されるヒープ領域である。ワークメモリ３７は、本来 Java 仮想マシン３８内に存在するが、図２１では、作図の便宜上、ワークメモリ３７を Java 仮想マシン３８上位層に記述している。ワークメモリ３７上の xlet プログラムには、EventListner や、JMF プレーヤインスタンスが含まれる。

Java 仮想マシン３８は、アプリケーションを構成する xlet プログラムをワークメモリ３７にロードして、xlet プログラムを解読し、解読結果に従った処理を実行する。上述したように xlet プログラムは、JMF プレーヤインスタンス生成を命じるメソッド、この JMF プレーヤインスタンスの実行を命じるメソッドを含むので、これらのメソッドで命じられた処理内容を実現するよう、下位層に対する制御を行う。JMF プレーヤインスタンス生成が命じられれば、Java 仮想マシン３８は、BD-ROM 上の YYYY.MPLS ファイルに関連付けられた JMF プレーヤインスタンスを得る。また、JMF プレーヤインスタンスにおける JMF メソッドの実行が命じられれば、この JMF メソッドを BD ミドルウェアに発行して、BD 再生装置が対応しているファンクションコールに置き換させる。そして置換後のファンクションコールを Playback Control Engine３２に発行する。

Event Listner Manager３９は、ユーザ操作により生じたイベント(キーイベ

ント)を解析し、イベントの振り分けを行う。図中の実線矢印◇1,◇2は、このEvent Listner Manager３９による振り分けを模式的に示す。START、STOP、SPEED等、xletプログラム内のEvent Listnerに登録されたキーイベントなら、BD-Jオブジェクトにより間接参照されているxletプログラムにかかるイベントを振り分ける。START、STOP、SPEEDは、JMFに対応したイベントであり、xletプログラムのEvent Listnerには、これらのキーイベントが登録されているので、本キーイベントによりxletプログラムの起動が可能になる。キーイベントがEvent Listner非登録のキーイベントである場合、本キーイベントをDefault Operation Manager４０に振り分ける。音声切り換え、アングル切り換え等、BD-ROM再生装置において生じるキーイベントには、Event Listnerに登録されていない多様なものがあり、これらのキーイベントが生じたとしても、漏れの無い処理を実行するためである。

Default Operation Manager４０は、xletプログラム内のEvent Listnerに登録されてないキーイベントがEvent Listner Manager３９から振り分けられると、そのEvent Listner非登録イベントに対応するファンクションコールをPlayback Control Engine３２に対して実行する。このDefault Operation Manager４０によるファンクションコールを模式的に示したのが、図中の矢印◇3である。尚、図２１においてEvent Listner非登録イベントはEvent Listner Manager３９、Default Operation Manager４０により振り分けられたが、Playback Control Engine３２がダイレクトにEvent Listner非登録イベントを受け取り、再生制御を行ってもよい(図中の◇4)。

(フローチャートの説明)

以上のアプリケーションマネージャ３６についての説明は、その概要に触れたに過ぎない。アプリケーションマネージャ３６の処理を更に詳しく示したのが図２２、図２３のフローチャートである。以降、これらのフローチャートを参照してアプリケーションマネージャ３６の処理手順についてより詳しく説明する。

図２２は、アプリケーションマネージャ３６による分岐時の制御手順を示す

図である。本フローチャートは、ステップＳ２～ステップＳ５がなす条件を満たすアプリケーション(アプリケーション x という)を、起動又は終了させるという処理である。

ステップＳ２は、分岐元タイトルで非起動だが、分岐先タイトルで生存していて、分岐先タイトルにおける起動属性が AutoRun 属性のアプリケーション x が存在するか否かの判定であり、もしあれば、ローカルメモリ２９に対するキャッシュセンスを行う。キャッシュセンスの結果、アプリケーション x がローカルメモリ２９上に有れば(ステップＳ７で Yes)、ローカルメモリ２９からワークメモリ３７にアプリケーション x を読み込む(ステップＳ８)。ローカルメモリ２９に無ければ、BD-ROM からローカルメモリ２９にアプリケーション x を読み込んだ上で、ローカルメモリ２９からワークメモリ３７にアプリケーション x を読み込む(ステップＳ９)。

ステップＳ３は、分岐元タイトルで起動中で、分岐先タイトルで非生存のアプリケーション x が存在するかどうかの判定である。もし存在するのなら、アプリケーション x をワークメモリ３７から削除して終了させる(ステップＳ１０)。

ステップＳ４は、分岐元 Suspend、分岐先 AutoRun 又は Persistent のアプリケーションが存在するか否かの判定である。もし存在するなら、アプリケーション x を Resume する(ステップＳ１１)。

ステップＳ５は、分岐元タイトルで起動中で、分岐先 Suspend のアプリケーションが存在するか否かの判定である。もし存在すれば、アプリケーション x を Suspend する(ステップＳ１２)。

個々のアプリケーションを終了させるにあたってのアプリケーションマネージャ３６の処理は、図２３に示すものとなる。図２３は、アプリケーション終了処理の処理手順を示すフローチャートである。本図は、終了すべき複数アプリケーションのそれぞれについて、ステップＳ１６～ステップＳ２０の処理を繰り返すループ処理になっている(ステップＳ１５)。本ループ処理においてアプリケーションマネージャ３６は起動中アプリケーションを終了するような terminate イベントを発行し(ステップＳ１６)、タイマセットして(ステップＳ

１７)、ステップＳ１８～ステップＳ２０からなるループ処理に移行する。このterminateイベントをEvent Listnerが受信すれば、対応するxletプログラムは、終了プロセスを起動する。終了プロセスが終了すれば、そのxletプログラムはワークメモリ３７から解放され、終了することになる。

ステップＳ１８～ステップＳ２０のループ処理の継続中、タイマはカウントダウンを継続する。本ループ処理においてステップＳ１８は、発行先アプリケーションが終了したか否かの判定であり、もし終了していればこのアプリケーションに対する処理を終える。ステップＳ１９は、タイマがタイムアウトしたか否かの判定であリ、タイムアウトすれば、ステップＳ２０において発行先アプリケーションをワークメモリ３７から削除してアプリケーションを強制終了する。

以上のモジュールマネージャ３４の処理を、図２４を参照しながら説明する。

図２４は、アプリケーション終了の過程を模式的に示した図である。本図における第１段目は、アプリケーションマネージャ３６を、第２段目は、３つのアプリケーションを示す。図２４の第２段目、左側のアプリケーションは、terminateイベントを受信して終了プロセスに成功したアプリケーションを示す。図２４の第２段目、中列のアプリケーションは、terminateイベントを受信したが終了プロセスに失敗したアプリケーションを示す。第２段目、右側のアプリケーションは、EventListnerが実装されていないので、terminateイベントを受信することができなかったアプリケーションを示す。

第１段目－第２段目間の矢印ep1,ep2は、アプリケーションマネージャによるterminateイベント発行を模式的に示し、矢印ep3は、終了プロセス起動を模式的に示している。

第３段目は、終了プロセス成功時における状態遷移後の状態であり、このアプリケーションは、自身の終了プロセスにより終了することになる。これらxletプログラムのように、所定の期間内に終了しないアプリケーションがあれば、アプリケーションマネージャ３６は、それらを強制的にワークメモリ３７から取り除く。第４段目は、アプリケーションマネージャ３６による強制終了を示す。かかる第４段目の強制終了を規定するのも、アプリケーションマネージャ

３６の１つの使命といえる。

以上のように本実施形態によれば、分岐元タイトルで起動しており、分岐先タイトルで生存していないアプリケーションは、自動的に終了させられるので、条件付き分岐により再生が複雑に進行する場合でも、再生装置におけるリソースの限界を越える数のアプリケーション立ち上げはなされ無い。分岐前後におけるアプリケーション動作を保証することができるので、デジタルストリームを再生させながら、アプリケーションを実行させるようなディスクコンテンツを多く頒布することができる。

(第２実施形態)

第１実施形態においてアプリケーションの生存区間は、タイトル時間軸と一致していたが、第２実施形態は、PL 時間軸の一部をアプリケーションの生存区間とすることを提案する。PL 時間軸の一部は、チャプターにより表現されるので、チャプターにて開始点、終了点を記述することにより、アプリケーションの生存区間を規定することができる。図２５（ａ）は、PL 時間軸上に生存区間を定めたアプリケーション管理テーブルを示す図である。図２５（ａ）においてアプリケーション管理テーブルには、3 つのアプリケーションが記述されており、このうち application#2 は、title#1 の Chapter#2 から Chapter#3 までが生存区間に指定され、起動属性に AutoRun が規定されている。このため application#2 は、図２５（ｂ）に示すように、Chapter#2 の始点で起動され、Chapter#3 の終点で終了することになる。

一方 application#3 は、title#1 の Chapter#4 から Chapter#6 までが生存区間に指定されている。このため application#3 は、図２５（ｂ）に示すように、図２５（ｂ）に示すように、Chapter#4 の始点で起動され、Chapter#6 の終点で終了することになる。

こうして記述されたアプリケーション管理テーブルに基づき、処理を行うため本実施形態に係るアプリケーションマネージャ３６は、PLmark により指示されるチャプター開始点に到達する度に、そのチャプター開始点から生存区間が始まるアプリケーションが存在するか否かを判定し、もし存在すればそのアプリケーションをワークメモリ３７にロードする。

同様に、チャプター開始点に到達する度に、そのチャプターの直前のチャプターで生存区間が終わるアプリケーションが存在するか否かを判定し、もし存在すればそのアプリケーションをワークメモリ３７から解放する。

チャプターという単位でアプリケーションの生存を管理すれば、アプリケーションの生存区間をより細かい精度で指定することができる。しかしディスクコンテンツには、時間軸の逆向がありうることに留意せねばならない。逆行とは、巻戻しにより時間軸を逆向きに進行することである。チャプターの境界でこの逆行と、進行とが繰り返されれば、ワークメモリへのロード、廃棄が何度もなされ、余分な読出負荷が生じる。そこで本実施形態では、アプリケーションの起動時期を、タイトルに入って Playback Control Engine３２による通常再生が開始された瞬間にしている。ここで PL の再生には、通常再生、トリック再生がある。トリック再生とは、早送り、巻戻し、SkipNext,SkipBack がある。かかる、早送り、巻戻し、SkipNext,SkipBack がなされている間、アプリケーション起動を開始せず、通常再生が開始されて初めて、アプリケーションを起動するのである。通常再生開始の瞬間を基準にすることにより、上述したような生存区間前後の行き来があった場合でも、アプリケーションの起動が必要以上に繰り返されることはない。尚、通常再生開始の瞬間を、アプリケーションの起動基準にするとの処理は、生存区間が title である場合にも、実行してよい。

以上のように本実施形態によれば、PL より小さい、チャプターの単位でアプリケーションの生存区間を規定することができるので、緻密なアプリケーション制御を実現することができる。

(第２実施形態の変更例)

図２５では、各アプリケーションに優先度が付与されている。この優先度は、0～255 の値をとり、アプリケーション間でリソースの使用が競合等が競合した場合、どちらのアプリケーションを強制的に終了させるか、また、どちらのアプリケーションからリソースを奪うかという処理をアプリケーションマネージャ３６が行うにあたって、判断材料になる。図２５の一例では、application#1 の優先度は 255,application#2、application#3 の優先度は 128 なので、

application#1－application#2 の競合時において、アプリケーションマネージャ３６は優先度が低い application#2 を強制終了するとの処理を行う。

（第３実施形態）

BD-ROM により供されるディスクコンテンツは、互いに分岐可能な複数タイトルから構成される。各タイトルは、１つ以上の PL と、この PL を用いた制御手順とからなるもの以外に、再生装置に対する制御手順のみからなる非 AV 系タイトルがある。本実施形態は、この非 AV 系タイトルについて説明する。

こうした非 AV 系タイトルのタイトルでは、どのようにタイトル時間軸を定めるのかが問題になる。図２６（ａ）は、PL 時間軸がら定まるタイトル時間軸を示す。この場合 PL 時間軸がタイトル時間軸になり、このタイトル時間軸上にアプリケーションの生存区間が定まる。この基準となる PL 時間軸がない場合、タイトル時間軸は図２６（ｂ）（ｃ）のように定めるべきである。

図２６（ｂ）は、メインとなるアプリケーションの生存区間から定まるタイトル時間軸を示す。メインアプリとは、タイトルにおいて起動属性が AutoRun に設定され、タイトル開始時に自動起動される唯一のアプリケーションであり、例えばランチャーアプリと呼ばれるものがこれにあたる。ランチャーアプリとは、他のアプリケーションを起動するアプリケーションプログラムである。

この図２６（ｂ）の考え方は、メインアプリが起動している限り、タイトル時間軸は継続していると考え、メインアプリが終了すれば、時間軸を終結させるというものである。図２６（ｃ）は、複数アプリケーションの生存区間から定まるタイトル時間軸を示す図である。タイトルの開始点で起動されるのは １つのアプリケーションであるが、このアプリケーションが他のアプリケーションを呼び出し、更にこのアプリケーションが別のアプリケーションを呼び出すとの処理が繰り返されるというケースがある。この場合、どれかのアプリケーションが起動している限り、タイトル時間軸は継続していると考え、どのアプリケーションも起動していない状態が到来すれば、そこでタイトル時間軸は終結するという考え方である。このように非 AV 系タイトルのタイトル時間軸を定めれば、AV タイトルであっても、非 AV 系タイトルであっても、タイトル時間軸の終結と同時に、所定のタイトルに分岐するという処理を画一的に行うこ

とができる。尚非AVタイトルにおけるタイトル時間軸は、AVタイトルと対比する上で、想定した架空の時間軸に過ぎない。故に再生装置は、非AVタイトルにおけるタイトル時間軸上を逆行したり、任意の位置に頭出しすることができない。

以上は本実施形態における記録媒体に対する改良である。続いて本実施形態における再生装置に対する改良について説明する。

上述したような手順でタイトル終了を行うため第３実施形態に係るアプリケーションマネージャ３６は、図２７に示すような処理で処理を行う。図２７は、タイトル再生時におけるアプリケーションマネージャ３６の処理手順を示すフローチャートである。本フローチャートは、タイトル再生中、ステップＳ２１～ステップＳ２３を繰り返すというループ構造になっている。

ステップＳ２１は、タイトルジャンプAPIが呼び出されたか否かの判定であり、呼び出されれば、ジャンプ先タイトルへの分岐をモジュールマネージャ３４に要求する(ステップＳ２７)。

ステップＳ２２は、タイトル内のアプリケーション呼出を担っているようなメインアプリが存在するか否かの判定であり、もし存在するなら、それの起動の有無を確認する(ステップＳ２５)。起動してなければ、”タイトルの終わり”であると解釈し、モジュールマネージャ３４に終結を通知する(ステップＳ２６)。

ステップＳ２３は、メインアプリがない場合に実行されるステップであり(ステップＳ２２でNo)、どのアプリケーションも起動してない状態かどうかを判定する。もしそうなら、同じく”タイトルの終わり”であると解釈し、モジュールマネージャ３４に終結を通知する(ステップＳ２６)。

以上のように本実施形態によれば、PL再生を伴わないタイトルであっとしても、アプリケーション実行中は分岐せず、アプリケーション実行が終了して初めて分岐するという処理が可能になる。

(第４実施形態)

本実施形態は、DVDと同様のメニュー制御をBD-ROM上で実現する場合の改良に関する。図２８（ａ）は、BD-ROMにより実現されるメニュー階層を

示す図である。本図におけるメニュー階層は、TopMenu を最上位に配し、この TopMenu から下位の TitleMenu、SubTitleMenu、AudioMenu を選択できる構造になっている。図中の矢印 sw1,2,3 は、ボタン選択によるメニュー切り換えを模式的に示す。TopMenu とは、音声選択、字幕選択、タイトル選択の何れを行うかを受け付けるボタン(図中のボタン sn1,sn2,sn3)を配置したメニューである。

TitleMenu とは、映画作品(title)の劇場版を選択するか、ディレクターズカット版を選択するか、ゲーム版を選択するか等、映画作品の選択を受け付けるボタンを配置したメニューである。AudioMenu とは、音声再生を日本語で行うか、英語で行うかを受け付けるボタンを配置したメニュー、SubTitleMenu とは、字幕表示を日本語で行うか、英語で行うかを受け付けるボタンを配置したメニューである。

こうした階層をもったメニューを動作させるための MOVIE オブジェクトを図２８（ｂ）に示す。図２８（ｂ）において MovieObject.bdmv には、FirstPlay OBJ、TopMenu OBJ、AudioMenu OBJ、SubTitleMenu OBJ が格納されている。

FirstPlay オブジェクト(FirstPlay OBJ)は、再生装置への BD-ROM のローディング時に自動的に実行される動的シナリオである。

TopMenu オブジェクト(TopMenu OBJ)は、TopMenu の挙動を制御する動的シナリオである。ユーザがメニューコールを要求した際、呼び出されるのはこの TopMenu オブジェクトである。TopMenu オブジェクトは、ユーザからの操作に応じて TopMenu 中のボタンの状態を変えるものや、ボタンに対する確定操作に応じて分岐を行う分岐コマンドを含む。この分岐コマンドは、TopMenu から TitleMenu、TopMenu から SubTitleMenu、TopMenu から AudioMenu というメニュー切り換えを実現するものである。

AudioMenu オブジェクト(AudioMenu OBJ)は、AudioMenu の挙動を制御する動的シナリオであり、ユーザからの操作に応じて AudioMenu 中のボタンの状態を変えるコマンドや、ボタンに対する確定操作に応じて音声設定を更新するコマンドを含む。

SubTitleMenu オブジェクト(SubTitleMenu OBJ)は、SubTitleMenu の挙動を制御する動的なシナリオであり、ユーザからの操作に応じて SubTitleMenu 中のボタンの状態を変えるコマンドや、ボタンに対する確定操作に応じて字幕設定用の PSR を更新するコマンドを含む。

TitleMenu オブジェクト(TitleMenu OBJ)は、TitleMenu の挙動を制御する動的シナリオであり、TitleMenu 中のボタンの状態を変えるものや、ボタンに対する確定操作に応じて分岐を行う分岐コマンドをふくむ。

これらのメニュー用 MOVIE オブジェクトにより、DVD で実現されているようなメニューの挙動を実現することができる。以上がメニュー制御に関連する MOVIE オブジェクトである。

図２９は、Index Table と、Index Table から各 Movie オブジェクトへの分岐とを模式化した図である。本図では左側に Index Table の内部構成を示している。本実施形態における Index Table には、FirstPLayINDEX、TopMenuINDEX, Audio MenuINDEX、Subtitle MenuINDEX、title MenuINDEX、title#1～#mINDEX、title#m+1～#nINDEX、title#0INDEX を含む。図中の矢印 bc1,2 は、Index Table から FirstPlayOBJ への分岐と,FirstPlayOBJ から TopMenu への分岐とを模式的に示し、矢印 bc3,4,5 は、TopMenu から TitleMenu、SubTitleMenu、AudioMenu への分岐を模式的に示している。矢印 bc6,7,8 は、TitleMenu から各 Movie オブジェクトへの分岐を模式的に示している。

FirstPLayINDEX 、 TopMenuINDEX, Audio MenuINDEX 、 Subtitle MenuINDEX、title MenuINDEX は、それぞれ、FirstPLayOBJ、TopMenuOBJ, Audio MenuOBJ、Subtitle MenuOBJ、title MenuOBJ についての Index であり、これらの識別子が記述される。

title#1～#mINDEX は、BD-ROM において 1 から m 番目にエントリーされている title についての Index であり、これら 1 から m までの title 番号の選択時において分岐先となる MOVIE オブジェクトの識別子(ID)が記述される。

title#m+1～#nINDEX は、BD-ROM において m+1 から n 番目にエントリーされている title についての Index であり、これら m+1 から n までの title

番号の選択時において分岐先となる BD-J オブジェクトの識別子(ID)が記述される。

title#0INDEX は、BD-J オブジェクトの強制終了時において分岐先になるべき Movie オブジェクト又は BD-J オブジェクトを規定する INDEX である。本実施形態では、TopMenuOBJ についての識別子が、この title#0INDEX に格納されている。

図３０（ａ）は、図２９のように Index Table が記述された場合における分岐を示す。Index Table がこのように記述されているので、ラベル title#1～title#m を分岐先とした分岐コマンドの実行時には、title#1Index～title#mIndex から Movie オブジェクト#1～#m の識別子が取り出される。ラベル title#m+1～title#n を分岐とした分岐コマンドの実行時には、title#m+1Index～title#nIndex から BD-J オブジェクト#m+1～#n の識別子が取り出される。BD-J オブジェクト#m+1～#n の識別子は、ファイル名を表す５桁の数値であるので、『00001.BD-J,00002.BD-J,00003.BD-J･･･』が取り出され、そのファイル名の動的シナリオがメモリに読み出されて、実行されることになる。これが Index Table を用いた分岐処理である。

図３０（ｂ）は、BD-J オブジェクト実行時の強制終了時における分岐を示す図である。強制終了時における分岐では、title#0Index から識別子が取り出されて、その識別子の動的シナリオが再生装置により実行される。この識別子が、トップメニュータイトルの識別子なら、アプリケーション強制終了時には、自動的にトップメニューOBJ が選択されることになる。

以上は本実施形態における記録媒体に対する改良である。続いて本実施形態における再生装置に対する改良について説明する。上述した記録媒体の改良に対応するため、再生装置内のモジュールマネージャ３４は図３１に示すような処理手順で処理を行う。図３１は、モジュールマネージャ３４の処理手順を示すフローチャートである。本フローチャートは、ステップＳ３１、ステップＳ３２からなるループ処理を構成しており、ステップＳ３１又はステップＳ３２のどちらかが Yes になった際、対応する処理を実行するものである。

ステップＳ３１は、タイトルジャンプAPIの呼び出しがあったか否かの判定である。もしタイトルジャンプAPIの呼び出しがあれば、分岐先ラベルであるタイトル番号jを取得し(ステップＳ３３)、Index Tableにおけるタイトル番号jのIndexから、IDjを取り出して(ステップＳ３４)、IDjのMovieオブジェクト又はBD-Jオブジェクトを、HDMVモジュール３３又はBD-Jモジュール３５に実行させる(ステップＳ３５)。

ステップＳ３２は、タイトル終了がアプリケーションマネージャ３６から通知されたか否かの判定であり、もし通知されれば(ステップＳ３２でYes)、トップメニュータイトルを構成するトップメニューOBJをHDMVモジュール３３又はモジュールマネージャ３４に実行させる(ステップＳ３６)。

以上のアプリケーションマネージャ３６によるアプリケーション強制終了の動作例を、図３２を参照しながら説明する。ここで再生すべきタイトルは、落下するタイル片を積み重ねるというゲームアプリを含む非AV系タイトルである。図３２の下段は、アプリケーションの生存区間からなるタイトル時間軸を示し、上段は、タイトル時間軸において表示される画像を示す。非AV系タイトルがゲームアプリである場合、このゲームアプリの生存区間において、図３２の上段左側のように、ゲームアプリの一画面が表示される。ゲームアプリにバグがあり、異常終了すると、アプリケーションマネージャ３６は図２３のフローチャートに従ってゲームアプリを強制終了させ、タイトルの終了をモジュールマネージャ３４に通知する。タイトル終了が通知されると、モジュールマネージャ３４はトップメニュータイトルに分岐する。そうすると、図３２の上段右側に示すような画像が表示され、ユーザの操作待ちになる。

以上のように本実施形態によれば、プログラムが含むが、デジタルストリームは含まないような非AV系タイトルの終了時においても、トップメニュータイトルに分岐するという制御が可能になる。これによりアプリケーションプログラムがエラー終了したとしても、ブラックアウトやハングアップの発生を回避することができる。

(第５実施形態)

BD-J モードにおいて、PL 再生との同期をどのように実現するかという改良に関する。図８（ｂ）の一例において JMF プレーヤインスタンスの再生を命じる JMF プレーヤインスタンス(A.play;)を Java 仮想マシン３８が解読した場合、Java 仮想マシン３８は PL 再生 API をコールして、コール直後に”サクセス”を示す応答をアプリケーションに返す。

Playback Control Engine３２は、PL 再生 API がコールされれば、PL 情報に基づく処理手順を実行する。PL が 2 時間という再生時間を有するなら、この 2 時間の間、上述した処理は継続することになる。ここで問題になるのは、Java 仮想マシン３８がサクセス応答を返す時間と、Playback Control Engine３２が実際に処理を終える時間とのギャップである。Java 仮想マシン３８は、イベントドリブンの処理主体であるためコール直後に再生成功か、再生失敗かを示す応答を返すが、Playback Control Engine３２による実際の処理終了は 2 時間経過後であるので、サクセス応答をアプリケーションに返す時間を基準にしたのでは、2 時間経過後にあたる処理終結を感知しえない。PL 再生において早送り、巻戻し、Skip が行われると、この 2 時間という再生期間は 2 時間前後に変動することになり、処理終結の感知は更に困難になる。

Playback Control Engine３２は、アプリケーションとスタンドアローンで動作するため、第３実施形態のような終了判定では、PL 再生の終了をタイトル終了と解釈することができない。そこで本実施形態では、アプリケーションが終了してようがいまいが、ワークメモリ３７に JMF プレーヤインスタンスがある限り、つまり、Presentation Engine３１の制御権を BD-J モジュール３５が掌握している間、Playback Control Engine３２から再生終結イベントを待つ。そして再生終結イベントがあれば、タイトルが終了したと解釈して、次のタイトルへの分岐を行うようモジュールマネージャ３４に通知する。こうすることにより、Playback Control Engine３２が PL 再生を終結した時点を、タイトルの終端とすることができる。

以降図３３～図３７のフローチャートを参照して、Playback Control Engine３２による具体的な制御手順を説明する。

図３３は、Playback Control Engine３２による PL 再生手順を示すフローチ

ャートである。この再生手順は、Presentation Engine３１に対する制御(ステップＳ４６)と、BD-ROM ドライブ１又は HDD１７に対する制御(ステップＳ４８)とを主に含む。本フローチャートにおいて処理対象たる PlayItem を PlayItem#x とする。本フローチャートは、カレント PL 情報(.mpls)の読み込みを行い(ステップＳ４１)、その後、ステップＳ４２～ステップＳ５０の処理を実行するというものである。ここでステップＳ４２～ステップＳ５０は、ステップＳ４９が Yes になるまで、カレント PL 情報を構成するそれぞれの PI 情報について、ステップＳ４３～ステップＳ５０の処理を繰り返すというループ処理を構成している。このループ処理において処理対象となる PlayItem を、PlayItem#x(PI#x)とよぶ。この PlayItem#x は、カレント PL の先頭の PlayItem に設定されることにより、初期化される(ステップＳ４２)。上述したループ処理の終了要件は、この PlayItem#x がカレント PL の最後の PlayItem になることであり(ステップＳ４９)、もし最後の PlayItem でなければ、カレント PL における次の PlayItem が PlayItem#x に設定される(ステップＳ５０)。

ループ処理において繰り返し実行されるステップＳ４３～ステップＳ５０は、PlayItem#x の Clip_information_file_name で指定される Clip 情報をシナリオメモリ２１に読み込み(ステップＳ４３)、PlayItem#x の In_time を、カレント Clip 情報の EPmap を用いて、I ピクチャアドレス u に変換し(ステップＳ４４)、PlayItem#x の Out_time を、カレント Clip 情報の EP_map を用いて、I ピクチャアドレス v に変換して(ステップＳ４５)、これらの変換で得られたアドレス v の次の I ピクチャを求めて、そのアドレスの１つ手前をアドレス w に設定し(ステップＳ４７)、そうして算出されたアドレス w を用いて、I ピクチャアドレス u からアドレス w までの TS パケットの読み出しを BD-ROM ドライブ１又は HDD１７に命じるというものである(ステップＳ４８)。

一方、Presentation Engine３１に対しては、カレント PLMark の mark_time_stamp から PlayItem#x の Out_time までの出力を命じる(ステップＳ４６)。以上のステップＳ４５～ステップＳ４８により、AVClip において、PlayItem#x により指示されている部分の再生がなされることになる

その後、PlayItem#x がカレント PL の最後の PI であるかの判定がなされる

(ステップＳ４９)。

PlayItem#x がカレント PL の最後の PI でなければ、カレント PL における次の PlayItem を、PlayItem#x に設定して(ステップＳ５０)、ステップＳ４３に戻る。以上のステップＳ４３～ステップＳ５０を繰り返すことにより、PL を構成する PI は順次再生されることになる。

図３４は、アングル切り換え手順及び SkipBack,SkipNext の手順を示すフローチャートである。本フローチャートは、図３３の処理手順と並行してなされるものであり、ステップＳ５１～Ｓ５２からなるループ処理を繰り返すというものである。本ループにおけるステップＳ５１は、アングル切り換えを要求する API が、Java 仮想マシン３８からコールされたか否かの判定であり、アングル切り換え API のコールがあれば、カレント Clip 情報を切り換えるという操作を実行する。

図３４のステップＳ５５は、判定ステップであり、PlayItem#x の is_multi_angles がオンであるか否かの判定を行う。is_multi_angles とは、PlayItem#x がマルチアングルに対応しているか否かを示すフラグであり、もしステップＳ５５が No であるならステップＳ５３に移行する。ステップＳ５５が Yes であるなら、ステップＳ５６～ステップＳ５９を実行する。ステップＳ５６～ステップＳ５９は、切り換え後のアングル番号を変数 y に代入して(ステップＳ５６)、PlayItem#x における y 番目の Clip_information_file_name で指定されている Clip 情報をシナリオメモリ２１に読み出し(ステップＳ５７)、カレント PTM を、カレント Clip 情報の EP_map を用いてＩピクチャアドレス u に変換し(ステップＳ５８)、PlayItem#x の Out_time を、カレント Clip 情報の EP_map を用いてＩピクチャアドレス v に変換する(ステップＳ５９)というものである。こうしてＩピクチャアドレス u,v を変化した後、ステップＳ４６に移行する。ステップＳ４６への移行により、別の AVClip から TS パケットが読み出されるので、映像内容が切り換わることになる。

一方、図３４のループにおけるステップＳ５２は、SkipBack/SkipNext を意味する API が Java 仮想マシン３８からコールされたか否かの判定であり、もしコールされれば、図３５のフローチャートの処理手順を実行する。図３５は、

SkipBack,SkipNextAPIがコールされた際の処理手順を示すフローチャートである。SkipBack,SkipNextを実行するにあたっての処理手順は多種多様なものである。ここで説明するのはあくまでも一例に過ぎないことに留意されたい。

ステップＳ６１は、PSRで示されるカレントPI番号、及び、カレントPTMを変換することにより、カレントMark情報を得る。ステップＳ６２は、押下されたのがSkipNextキーであるか、SkipBackキーであるかの判定であり、SkipNextキーであるならステップＳ６３において方向フラグを+1に設定し、SkipBackキーであるならステップＳ６４において方向フラグを-1に設定する。

ステップＳ６５は、カレントPLMarkの番号に方向フラグの値を足した番号を、カレントPLMarkの番号として設定する。ここでSkipNextキーであるなら方向フラグは+1に設定されているのでカレントPLMarkはインクリメントされることになる。SkipBackキーであるなら方向フラグは-1に設定されているので、カレントPLMarkはデクリメントされることになる。

ステップＳ６６では、カレントPLMarkのref_to_PlayItem_Idに記述されているPIを、PlayItem#xに設定し、ステップＳ６７では、PlayItem#xのClip_information_file_nameで指定されるClip情報を読み込む。ステップＳ６８では、カレントClip情報のEP_mapを用いて、カレントPLMarkのmark_time_stampを、Ｉピクチャアドレスuに変換する。一方ステップＳ６９では、PlayItem#xのOut_timeを,カレントClip情報のEP_mapを用いて,Ｉピクチャアドレスvに変換する。ステップＳ７０は、カレントPLMarkのmark_time_stampからPlayItem#xのOut_timeまでの出力をPresentation Engine３１に命じた上で、図３３のステップＳ４７に移行する。こうしてＩピクチャアドレスu,vを変化して、別の部分の再生を命じた上でステップＳ４７への移行するので､別のAVClipからTSパケットが読み出されることになり、映像内容が切り換えが実現する。

図３６は、Presentation Engine３１による処理手順の詳細を示すフローチャートである。本フローチャートは、ＩピクチャのPTSをカレントPTMに設定した後で(ステップＳ７１)、ステップＳ７２～ステップＳ７７からなるループ処理を実行するものである。

続いてステップＳ７２～ステップＳ７７におけるループ処理について説明する。このループ処理は、カレント PTM にあたるピクチャ、オーディオの再生出力と、カレント PTM の更新とを繰り返すものである。本ループ処理におけるステップＳ７６は、ループ処理の終了要件を規定している。つまりステップＳ７６は、カレント PTM が PI#x の Out_time であることをループ処理の終了要件にしている。

ステップＳ７３は、早送り API、又は、早戻し API が Java 仮想マシン３８からコールされたか否かの判定である。もしコールされれば、ステップＳ７８において早送りか早戻しかの判定を行い、早送りであるなら、次のＩピクチャの PTS をカレント PTM に設定する(ステップＳ７９)。このようにカレント PTM を、次のＩピクチャの PTS に設定することで、1 秒飛びに AVClip を再生してゆくことができる。これにより、2 倍速等で AVClip は順方向に早く再生されることになる。早戻しであるなら、カレント PTM が PlayItem#x の Out_time に到達したかを判定する(ステップＳ８０)。もし到達してないのなら、1 つ前のＩピクチャの PTS をカレント PTM に設定する(ステップＳ８１)。このように読出先アドレスＡを、1 つ前のＩピクチャに設定することで、AVClip を後方向に 1 秒飛びに再生してゆくことができる。これにより、2 倍速等で AVClip は、逆方向に再生されることになる。尚、早送り、巻戻しを実行するにあたっての処理手順は多種多様なものである。ここで説明するのはあくまでも一例に過ぎないことに留意されたい。

ステップＳ７４は、メニューコール API がコールされたか否かの判定であり、もしコールされれば、現在の再生処理をサスペンドして(ステップＳ８２)、メニュー処理用のメニュープログラムを実行する(ステップＳ８３)。以上の処理により、メニューメニューコールがなされた場合は、再生処理を中断した上で、メニュー表示のための処理が実行されることになる。

ステップＳ７５は、sync_PlayItem_id により、PlayItem#x を指定した SubPlayItem#y が存在するか否かの判定であり、もし存在すれば、図３７のフローチャートに移行する。図３７は、SubPlayItem の再生手順を示すフローチャートである。本フローチャートでは、先ずステップＳ８６において、カレン

トPTMはSubPlayItem#yのsync_start_PTS_of_playItemであるか否かを判定する。もしそうであれば、ステップＳ９３においてSubPlayItem#yに基づく再生処理を行うようPlayback Control Engine３２に通知する。

図３７のステップＳ８７～ステップＳ９２は、SubPlayItem#yに基づく再生処理を示すフローチャートである。

ステップＳ８７では、SubPlayItem#yのClip_information_file_nameで指定されるClip情報を読み込む。ステップＳ８８では、カレントClip情報のEP_mapを用いて、SubPlayItem#yのIn_timeを、アドレスαに変換する。一方ステップＳ８９では、SubPlayItem#yのOut_timeを,カレントClip情報のEP_mapを用いて、アドレスβに変換する。ステップＳ９０は、SubPlayItem#yのIn_timeからSubPlayItem#yのOut_timeまでの出力をデコーダに命じる。これらの変換で得られたアドレスβの次のＩピクチャを求めて、そのアドレスの１つ手前をアドレスγに設定し(ステップＳ９１)、そうして算出されたアドレスγを用いて、SubClip#zにおけるアドレスαからアドレスγまでのTSパケットの読み出しをBD-ROMドライブ１又はHDD１７に命じるというものである(ステップＳ９２)。

また図３３に戻ってPlayback Control Engine３２の処理の説明の続きを行う。ステップＳ５３はPresentation Engine３１による再生制御が完了したかの判定であり、最後のPlayItem#xに対して、図３６のフローチャートの処理が行われている限り、ステップＳ５３がNoになる。図３６のフローチャートの処理が終了して初めて、ステップＳ５３はYesになりステップＳ５４に移行する。ステップＳ５４は、Java仮想マシン３８への再生終結イベントの出力であり、この出力により、2時間という再生時間の経過をJava仮想マシン３８は知ることができる。

以上が本実施形態におけるPlayback Control Engine３２、Presentation Engine３１の処理である。続いて本実施形態におけるアプリケーションマネージャ３６処理手順について説明する。図３８は、第５実施形態に係るアプリケーションマネージャ３６の処理手順を示すフローチャートである。

図３８のフローチャートは、図２７のフローチャートを改良したものである。

その改良点は、ステップＳ２１－ステップＳ２２間にステップＳ２４が追加され、このステップＳ２４がYesになった際、実行されるステップＳ１０１が存在する点である。

ステップＳ２４は、JMFプレーヤインスタンスがワークメモリ３７に存在するか否かの判定であり、もし存在しなければステップＳ２２に移行する。存在すれば、ステップＳ１０１に移行する。ステップＳ１０１は、Playback Control Engine３２から再生終結イベントが出力されたか否かの判定であり、もし出力されれば、ワークメモリ中のJavaプレーヤインスタンスを消滅させた上で(ステップＳ１０２)、タイトル終了をモジュールマネージャ３４に通知する(ステップＳ２６)。通知されねば、ステップＳ２１～ステップＳ２４からなるループ処理を繰り返す。

以上のフローチャートにおいて、ワークメモリ３７にJMFプレーヤインスタンスが存在する限り(ステップＳ２４でYes)、ステップＳ２２、ステップＳ２３はスキップされる。そのため、たとえ全てのアプリケーションが終了したとしてもタイトルは継続中と解釈される。

以上のように本実施形態によれば、2時間という再生時間の経過時点をアプリケーションマネージャ３６は把握することができるので、PL再生の終了条件にメニューを表示して、このメニューに対する操作に応じて他のタイトルに分岐するという制御を実現することができる。

(第６実施形態)

第６実施形態は、BD-Jオブジェクトにデータ管理テーブルを設ける改良に関する。

データ管理テーブル(DMT)は、そのタイトル時間軸においてローカルメモリ２９上にロードすべきJavaアーカイブファイルを、読込属性と、読込優先度とに対応づけて示すテーブルである。"ローカルメモリ２９における生存"とは、そのアプリケーションを構成するJavaアーカイブファイルがローカルメモリ２９から読み出され、Java仮想マシン３８内のワークメモリ３７への転送が可能になっている状態をいう。図３９は、データ管理テーブルの一例を示す図である。本図に示すようにデータ管理テーブルは、アプリケーションの『生存区

間』と、その生存区間をもったアプリケーションを識別する『applicationID』と、そのアプリケーションの『読込属性』と、『読込優先度』とを示す。

上述したようにアプリケーション管理テーブルには、生存区間という概念があり、データ管理テーブルにも同じ生存区間という概念がある。アプリケーション管理テーブルと同じ概念を、データ管理テーブルに設けておくというのは一見無駄のように思えるがこれには意図がある。

図４０は、BD-J オブジェクトが想定している実行モデルを示す図である。本図における実行モデルは、BD-ROM、ローカルメモリ２９、Java 仮想マシン３８からなり、BD-ROM、ローカルメモリ２９、ワークメモリ３７という三者の関係を示す。矢印 my1 は、BD-ROM→ローカルメモリ２９間の読み込みを示し、矢印 my2 は、ローカルメモリ２９→ワークメモリ３７間の読み込みを示す。矢印上の注釈は、これらの読み込みが、どのようなタイミングでなされるかを示す。注釈によると、BD-ROM→ローカルメモリ２９間の読み込みは、いわゆる" 先読み" であり、アプリケーションが必要となる以前の時点に行われねばならない。

また注釈によると、ローカルメモリ２９→ワークメモリ３７間の読み込みは、アプリケーションが必要になった際になされることがわかる。" 必要になった際" とは、アプリケーションの生存区間が到来した時点(1)、アプリケーションの呼出が他のアプリケーション又はアプリケーションマネージャ３６から指示された時点(2)を意味する。

矢印 my3 は、ワークメモリ３７におけるアプリケーションの占有領域の解放を示し、矢印 my4 は、ローカルメモリ２９におけるアプリケーションの占有領域の解放を示す。矢印上の注釈は、これらの読み込みが、どのようなタイミングでなされるかを示す。注釈によると、ワークメモリ３７上の解放は、アプリケーション終了と同時になされることがわかる。一方ローカルメモリ２９上の解放は、Java 仮想マシン３８にとって必要でなくなった時点でなされる。この必要でなくなった時点とは、" 終了時点" ではない。" 終了した上、再起動の可能性もない時点" であること、つまり該当する title が終了した時点を意味する。上述した読込・解放のうち、ワークメモリ３７における解放時点は、アプリケー

ション管理テーブルにおける生存区間から判明する。しかし”アプリケーションが必要となる以前の時点”、”終了した上、再起動の可能性もない時点”については、規定し得ない。そこで、オーサリング段階において、かかる時点をディスクコンテンツ全体の時間軸上で規定しておくため、本実施形態では各アプリケーションが生存している区間を、アプリケーション管理テーブルとは別に、データ管理テーブルに記述するようにしている。つまり”アプリケーションが必要となる以前の時点”をデータ管理テーブルにおける生存区間の始点と定義し、”終了した上、再起動の可能性もない時点”をデータ管理テーブルの終点と定義することにより、上述したローカルメモリ２９上の格納内容の遷移をオーサリング時に規定しておくことができる。これがデータ管理テーブルの記述意義である。

データ管理テーブルによるローカルメモリ２９生存区間の記述について説明する。ここで制作しようとするディスクコンテンツは３つのタイトル(title#1、title#2、title#3)からなり、これらタイトルの時間軸において、図４１（ｂ）に示すようなタイミングで、ローカルメモリ２９を使用したいと考える。この場合、title#1 時間軸の開始点において application#1、application#2 を構成する Java アーカイブファイルをローカルメモリ２９に読み込み、title#1 時間軸の継続中、application#1、application#2 をローカルメモリ２９に常駐させておく。そして title#2 時間軸の始点で、application#1 を構成する Java アーカイブファイルをローカルメモリ２９から解放して、代わりに application#3 を構成する Java アーカイブファイルをローカルメモリ２９に読み込んで、常駐させるというものである(以降、アプリケーションを構成する Java アーカイブファイルは、アプリケーションと同義に扱う。)。この場合のデータ管理テーブルの記述は、図４１（ａ）の通りであり、アプリケーションの applicationID を、その生存区間に対応づけて記述することで、ローカルメモリ２９に常駐すべきアプリケーションを表現する。図４１（ａ）では、application#1 の applicationID が title#1 と対応づけられて記述されており、application#2 の applicationID は title#1、title#2 と対応づけられ、application#3 の applicationID は title#3 と対応づけられて記述されていることがわかる。こうすることで、ローカルメ

モリ２９占有の時間的遷移がオーサリング担当者により規定されることになる。

データ管理テーブル、アプリケーション管理テーブルの組合せとしては、アプリケーション管理テーブルに規定する生存区間は、細かい再生単位にし、データ管理テーブルに規定する生存区間は、大まかな再生単位にすることが望ましい。大まかな再生単位には、タイトル、PL といった非シームレスな再生単位が望ましい。一方、細かい再生単位としては、PL 内のチャプターというようにシームレスな再生単位が望ましい。アプリケーションの生存区間をタイトル毎、PL 毎に定めれば、アプリケーションはローカルメモリ２９上に存在するので、そのタイトルの再生中においてアプリケーションは何時でも取り出せる状態になる。そうであれば、アプリケーションの生存区間を細かく定めたとしても、アプリケーションを即座に、仮想マシン上のワークメモリに読み出すことができるので、アプリケーションの起動･終了が頻繁になされたとしても、スムーズなアプリケーション実行を実現することができる。

次に、読込属性について説明する。

図２において Java アーカイブファイルは、AVClip とは別の記録領域に記録されることを前提にしていた。しかしこれは一例に過ぎない。Java アーカイブファイルは、BD-ROM において AVClip が占める記録領域に埋め込まれることがある。この埋め込みの態様には、カルーセル化、インターリーブユニット化という２種類がある。

ここで”カルーセル化”とは、対話的な放送の実現のために同一内容を繰り返しするという放送方式に変換することである。BD-ROM は、放送されたデータを格納するものではないが、本実施形態では、カルーセルの放送形式に倣って JAVA アーカイブファイルを格納するようにしている。図４２は、カルーセル化による Java アーカイブファイル埋め込みを示す図である。第１段目は、AVClip 中に埋め込む Java アーカイブファイルであり、第２段目は、セクション化を示す。第３段目は、TS パケット化、第４段目は、AVClip を構成する TS パケット列を示す。こうしてセクション化、TS パケット化されたデータ(図中の”D”)が、AVClip に埋め込まれるのである。カルーセルにより AVClip に多重化された Java アーカイブファイルは、読み出すにあたって、低帯域で読

み出されることになる。この低帯域での読み出しは、概して 2～3 分というように長期間を要するため、再生装置は Java アーカイブファイルを 2～3 分をかけて読み込むことになる。

図４３は、インターリーブ化による Java アーカイブファイル埋め込みを示す図である。第１段目は、埋め込まれるべき AVClip、第２段目は、AVClip にインターリーブ化された Java アーカイブファイル、第３段目は、BD-ROM の記録領域における AVClip 配置である。本図に示すように、ストリームに埋め込まれるべき Java アーカイブファイルは、インターリーブ化され、AVClip を構成する XXXXX.m2ts を構成する分割部分(図中の AVClip2/4,3/4)の合間に記録される。インターリーブ化により AVClip に多重化された Java アーカイブファイルは、カルーセル化と比較して、高い帯域で読み出されることになる。この高い帯域での読み出しであるため、再生装置は Java アーカイブファイルを比較的短期間に読み込むことになる。

カルーセル化･インターリーブ化された Java アーカイブファイルは、プリロードされるのではない。BD-ROM における AVClip の記録領域のうち、カルーセル化･インターリーブ化された Java アーカイブファイルが埋め込まれた部分に、現在の再生時点が到達した際、再生装置のローカルメモリ２９にロードされる。Java アーカイブファイルの記録態様には、図２に示すものの他に、図４２、図４３（ａ）に示すものがあるので、読込属性は、図４３（ｂ）に示すように、設定されうる。図４３（ｂ）に示すように、読込属性は、タイトル再生に先立ち、ローカルメモリ２９に読み込まれる旨を示す" Preload" と、タイトル再生中に、カルーセル化方式で読み込まれる旨を示す" Load.Carousel" と、タイトル再生中に、インターリーブ化方式で読み込まれる旨を示す" Load.InterLeave" とがある。読込属性には、カルーセル化されているか、インターリーブ化されているかが添え字で表現されているが、これを省略してもよい。

データ管理テーブルにおける生存区間の具体的な記述例について、図４４を参照しながら説明する。図４４（ａ）は、データ管理テーブルの一例を示す図である。図４４（ｂ）は、かかるデータ管理テーブルの割り当てによるローカ

ルメモリ２９の格納内容の変遷を示す図である。本図は、縦軸方向にローカルメモリ２９における占有領域を示し、横軸を、１つのタイトル内の PL 時間軸としている。データ管理テーブルにおいて application#1 は、１つのタイトル内の PL 時間軸全体を生存区間とするよう記述されているので、このタイトルの Chapter#1～Chapter#5 においてローカルメモリ２９内の領域を占有することになる。データ管理テーブルにおいて application#2 は、タイトル内の PL#1 における Chapter#1～Chapter#2 を生存区間とするよう記述されているので、このタイトルの Chapter#1～Chapter#2 においてローカルメモリ２９内の領域を占有することになる。データ管理テーブルにおいて application#3 は、タイトル内の PL#1 における Chapter#4～Chapter#5 を生存区間とするよう記述されているので、このタイトルの Chapter#4～Chapter#5 においてローカルメモリ２９内の領域を占有することになる。以上で、データ管理テーブルにおける生存区間についての説明を終える。

続いて読込優先度について説明する。読込優先度とは、ローカルメモリ２９への読み込みに対する優劣を決める優先度である。読込優先度には複数の値がある。2 段階の優劣を設けたい場合、Mandatory を示す値、optional を示す値を読込優先度に設定する。この場合、Mandatory は高い読込優先度を意味し、optional は、低い読込優先度を意味する。3 段階の優劣を設けたい場合、Mandatory を示す値、optional:high、optional:low を示す値を読込優先度に設定する。Mandatory は、最も高い読込優先度を示し、optional:high は、中程度の読込優先度、optional:low は、最も低い読込優先度を示す。データ管理テーブルにおける読込優先度の具体的な記述例について、図４５（ａ）（ｂ）を参照しながら説明する。この具体例で、想定しているローカルメモリ２９のメモリ規模は、図４５（ａ）に示すようなものである。図４５（ａ）は、新旧再生装置におけるローカルメモリ２９のメモリ規模を対比して示す図である。矢印 mk1 は旧再生装置におけるメモリ規模を、矢印 mk2 は新再生装置におけるメモリ規模をそれぞれ示す。この矢印の対比から、新再生装置におけるローカルメモリ２９のメモリ規模は、旧再生装置のそれと比較して、三倍以上である状

態を想定している。このようにメモリ規模にバラツキがある場合、アプリケーションは、図４５に示すような２つのグループに分類される。１つ目は、どのようなメモリ規模であっても読み込むんでおくべきアプリケーション(#1,#2)である。２つ目は、旧再生装置での読み込みは望まないが、新再生装置での読み込みは希望するアプリケーション(#3,#4)である。読み込もうとするアプリケーションが、これら２つのグループに分類されれば、前者に帰属するアプリケーションに、読込優先度=Mandatory を設定し、後者に属するアプリケーションに、読込優先度=Optional を設定する。図４５（ｂ）は、読込優先度が設定されたデータ管理テーブルの一例を示す図である。データ管理テーブルをこのように設定した上で、application#1～application#4 を BD-ROM に記録すれば、あらゆるメモリ規模の再生装置での再生を保証しつつも、メモリ規模が大きい再生装置では、より大きなサイズのデータを利用したアプリケーションを再生装置に再生させることができる。

以上は本実施形態における記録媒体に対する改良である。続いて本実施形態における再生装置に対する改良について説明する。上述した記録媒体の改良に対応するため、アプリケーションマネージャ３６は図４６に示すような処理手順で処理を行う。

図４６は、アプリケーションマネージャ３６によるプリロード制御の処理手順を示す図である。本フローチャートは、再生すべきタイトルにおけるデータ管理テーブルを読み込み(ステップＳ１１１)、データ管理テーブルにおいて最も高い読込優先度をもちつつ、applicationID が最も小さいアプリケーションをアプリケーションｉにした上で(ステップＳ１１２)、ステップＳ１１３、ステップＳ１１４の判定を経た上で、アプリケーションｉをローカルメモリ２９にプリロードする(ステップＳ１１５)という処理を、ステップＳ１１６が No 及びステップＳ１１７が No と判定されるまで、繰り返すというループ処理を構成している。

ステップＳ１１３は、アプリケーションｉの読込属性がプリロードであるか否かの判定であり、ステップＳ１１４は、アプリケーションの読込優先度が=Mandatory であるか Optional であるかの判定である。ステップＳ１１３に

おいてプリロードと判定され、ステップＳ１１４において読込優先度がMandatoryと判定されれば、アプリケーションはローカルメモリ２９にプリロードされることになる(ステップＳ１１５)。もしステップＳ１１３において読込属性がロードであると判定されれば、ステップＳ１１４～ステップＳ１１５はスキップされることになる。

ループ処理の終了要件を規定する２つのステップのうちステップＳ１１６は、applicationIDが次に高く、アプリケーションｉと同一読込優先度のアプリケーションｋが存在するか否かを判定するものである。そのようなアプリケーションｋが存在するなら、そのアプリケーションｋをアプリケーションｉにする(ステップＳ１１９)。

ループ処理の終了要件を規定する２つのステップのうちステップＳ１１７は、データ管理テーブルにおいて次に低い読込優先度をもつアプリケーションが存在するか否かの判定であり、もし存在すれば、その次に低い読込優先度をもつアプリケーションのうち、最も小さいapplicationIDをアプリケーションｋを選んで(ステップＳ１１８)、そのアプリケーションｋをアプリケーションｉにする(ステップＳ１１９)。これらステップＳ１１６、ステップＳ１１７がYesになっている限り、上述したステップＳ１１３～ステップＳ１１５の処理は繰り返されることになる。ステップＳ１１６、ステップＳ１１７において、該当するアプリケーションが無くなれば本フローチャートの処理は終了することになる。

ステップＳ１２０～ステップＳ１２３は、ステップＳ１１４において読込優先度＝Optionalであると判定された場合に、実行される処理である。

ステップＳ１２０は、同じapplicationIDをもち、読込優先度が高いアプリケーションｊが存在するか否かの判定である。

ステップＳ１２１は、ローカルメモリ２９の残り容量がアプリケーションｉのサイズを上回るか否かを判定するステップである。ステップＳ１２０がNo、ステップＳ１２１がYesである場合、ステップＳ１１５においてアプリケーションｉがローカルメモリ２９にプリロードされることになる。ステップＳ１２０がNo、ステップＳ１２１がNoである場合、アプリケーションｉはローカル

メモリ２９にプリロードされずそのままステップＳ１１６に移行することになる。

こうしておくと、読込優先度＝Optionalのデータは、ステップＳ１２０－ステップＳ１２１の判定がYesにならないと、ローカルメモリ２９へのプリロードがなされない。メモリ規模が小さい旧再生装置は、2～3個のアプリケーションを読み込んだ程度で、ステップＳ１２１の判定はNoになるが、メモリ規模が大きい新再生装置は、更に多くのアプリケーションを読み込んだとしても、ステップＳ１２１の判定はNoにならない。以上のように、旧再生装置では、ローカルメモリ２９にMandatoryのアプリケーションのみが読み込まれ、新再生装置には、Mandatoryのアプリケーションと、Optionalのアプリケーションとが読み込まれることになる。

ステップＳ１２２は、ステップＳ１２０においてYesと判定された場合に実行されるステップである。同じapplicationIDをもち、読込優先度が高いアプリケーションｊがローカルメモリ２９上に存在する場合、ローカルメモリ２９の残り容量と、アプリケーションｊのサイズとの和が、アプリケーションｉのサイズを上回るか否かを判定し(ステップＳ１２２)、もし上回れば、アプリケーションｉを用いてローカルメモリ２９上のアプリケーションｊを上書きすることによりプリロードする(ステップＳ１２３)。下回る場合は、アプリケーションｉはローカルメモリ２９にプリロードされずそのままステップＳ１１６に移行することになる。

ステップＳ１１５、ステップＳ１２３による読込処理の一例を、図４７（ａ）を参照しながら説明する。図４７（ａ）は、この具体例が想定しているデータ管理テーブルの一例を示す図である。本図における３つのアプリケーションは、それぞれ３つのファイルに格納されており、applicationIDは同じであるが(applicationID=1)、読込優先度は互いに異なる(mandatory,optional:high,optional:low)。こうしたデータ管理テーブルが処理対象であると、ステップＳ１１５により、読込優先度＝Mandatoryのアプリケーションはローカルメモリ２９に読み込まれる。しかし読込優先度＝Optionalのアプリケーションについては、ステップＳ１２０～ステップＳ１２２の判定

を経た上で、ステップＳ１２３において読み込まれる。ステップＳ１１５と違いステップＳ１２３では、既にローカルメモリ２９にある同じ applicationID のアプリケーションを上書きしてゆくよう、プリロードがなされるので、複数アプリケーションのうち１つが排他的に、ローカルメモリ２９にロードされることになる。

i)読込優先度＝mandatory のアプリケーションを読み込んだ後、読込優先度＝optional:high のアプリケーションを読み込むにあたって、ステップＳ１２２が No と判定されればれば、読込優先度＝mandatory のアプリケーションがローカルメモリ２９に残ることになる。読込優先度＝mandatory のアプリケーションを読み込んだ後、読込優先度＝optional:high のアプリケーションを読み込むにあたって、ステップＳ１２２が Yes と判定されれば、読込優先度＝optional:high のアプリケーションにより、読込優先度＝mandatory のアプリケーションは上書きされ、読込優先度＝optional:high のアプリケーションがローカルメモリ２９に残ることになる。

ii)読込優先度＝optional:high のアプリケーションを読み込んだ後、読込優先度＝optional:low のアプリケーションを読み込むにあたって、ステップＳ１２２が No と判定されればれば、読込優先度＝Mandatory のアプリケーションがローカルメモリ２９に残ることになる。読込優先度＝optional:high のアプリケーションを読み込んだ後、読込優先度＝optional:low のアプリケーションを読み込むにあたって、ステップＳ１２２が Yes と判定されれば、読込優先度＝optional:low のアプリケーションにより、読込優先度＝optional:high のアプリケーションは上書きされ(ステップＳ１２３)、読込優先度＝optional:low のアプリケーションがローカルメモリ２９に残ることになる。

ローカルメモリ２９の容量が許す限り、ローカルメモリ２９上のアプリケーションを上書きしてゆくとの処理が繰り返されるので、ローカルメモリ２９の格納内容は、図４７（ｂ）に示すように、mandatory＝optional:high＝>optional:low と遷移してゆくことになる。メモリ規模に応じて、サイズが異なる Java アーカイブファイルをローカルメモリ２９にロードすることができる

ので、メモリ規模が小さい再生装置については、必要最小限の解像度をもったサムネール画像を有する Java アーカイブファイルを、メモリ規模が中程度の再生装置については、中程度の解像度をもった SD 画像を有する Java アーカイブファイルを、メモリ規模が大規模である再生装置については、高解像度をもった HD 画像を有する Java アーカイブファイルをローカルメモリ２９にロードすることができる。かかるロードにより、メモリ規模に応じて解像度が異なる画像を表示させることができ、オーサリング担当者によるタイトル制作の表現の幅が広がる。

図４８は、データ管理テーブルを参照した読取処理の具体例を示す図である。本図における 2 つのアプリケーションは、同じ applicationID(application#3)が付与された 2 つのアプリケーションを示す図である。そのうち一方は、AVClip 中に埋め込まれていて、読込優先度が mandatory に設定されている。他方は、AVClip とは別ファイルに記録されていて、読込優先度が Optional に設定されている。前者のアプリケーションは、AVClip に埋め込まれているので、その埋込部分にあたる生存区間が、生存区間(title#1:chapter#4～#5)として記述されている。これらのアプリケーションのうち application#2、application#3 には、ロードを示す読込属性が付与されている。application#2 は Chapter#1～Chapter#2 を生存区間にしており、application#3 は Chapter#4～Chapter#5 を生存区間にしているので、タイトル時間軸においてどちらか一方が排他的にローカルメモリ２９上に常駐することになる。図４８（ｂ）は、タイトル時間軸上の別々の時点において、排他的に格納される application#2、application#3 を示す図である。これは必要最低限のメモリ規模しかもたない再生装置での再生を念頭に置いた配慮である。こうした内容のデータ管理テーブルが処理対象であるとアプリケーションマネージャ３６は、上述した図４６のフローチャートによりメモリ規模に応じて異なる処理を行う。

後者のアプリケーションは、読込優先度＝ロードであるので、ローカルメモリ２９にロードされる。かかる処理により、Mandatory なメモリ規模さえあれば、アプリケーションマネージャはデータをローカルメモリ２９にロードする

ことができる。ここで問題になるのは、メモリ規模が大きい再生装置による読み込み時である。メモリ規模が大きいにも拘らず、Chapter#4～Chapter#5 に到達するまで application#3 を読み込めないというのは、メモリ規模の無駄になる。そこで本図のデータ管理テーブルには、同じ application#3 にプリロードを示す読込属性を付与して BD-ROM に記録しておき、これらに同じ applicationID を付与している。

前者のアプリケーションは、読込優先度＝Optional であるので、ステップＳ１２１が Yes になった場合に限り、プリロードされる(ステップＳ１１５)。こうすることで、メモリ規模が大きい再生装置は、title#1、Chapter#4～Chapter#5 の到達を待つことなく、AVClip に埋め込まれているのと同じアプリケーションをローカルメモリ２９にロードすることができるのである(図４８（ｃ）)。

以上がプリロード時における処理である。続いてロード時における処理手順について説明する。

図４９は、データ管理テーブルに基づくロード処理の処理手順を示す図である。本フローチャートは、ステップＳ１３１～ステップＳ１３３からなるループ処理を、タイトル再生が継続されている間、繰り返すというものである。

ステップＳ１３１は、AutoRun を示す起動属性を有したアプリケーションの生存区間が到来したか否かの判定である。もし到来すれば、AutoRun を示す起動属性を有したアプリケーションをアプリケーション q にして(ステップＳ１３４)、アプリケーション q を起動する旨の起動指示を Java 仮想マシン３８に発行して、アプリケーション q をローカルメモリ２９からワークメモリ３７に読み出させる(ステップＳ１３５)。

ステップＳ１３３は、タイトル内 PL の再生が全て終了したかの判定である。この判定は、第５実施形態に示したように、Playback Control Engine３２からの再生終結イベントがあったか否かでなされる。もし終了すれば、本フローチャートの処理を終了する。

ステップＳ１３２は、起動中アプリケーションからの呼出があったか否かの判定である。もしあれば、呼出先アプリケーションをアプリケーション q にし

て(ステップＳ１３６)、 現在の再生時点は、アプリケーション管理テーブルにおけるアプリケーション q の生存区間であるか否かを判定する(ステップＳ１３７)。もし生存区間でなければ、起動失敗を表示して(ステップＳ１４８)、ステップＳ１３１～ステップＳ１３３からなるループ処理に戻る。生存区間であれば、図５０のフローチャートに従い、ロード処理を行う。

図５０におけるステップＳ１３８は、現在の再生時点がデータ管理テーブルにおけるアプリケーションｑの生存区間であるか否かを示す判定である。もし生存区間でなければ、アプリケーションｑはローカルメモリ２９にロードすることができない。この場合、アプリケーション q を起動する旨の起動指示をJava 仮想マシン３８に発行し、ローカルメモリ２９を介することなく、直接アプリケーションｑを BD-ROM からワークメモリ３７に読み出させる。この場合アプリケーションを読み出すためのヘッドシークが発生するから、PL 再生は中断することになる(ステップＳ１４５)。

もし生存区間であれば、ステップＳ１３９において、アプリケーションには読込属性が付加されているか否かを判定する。読込属性がないということは、アプリケーションｑは、カルーセル化、若しくはインターリーブ化されていないことを意味する。しかし読込属性が付加されていなくても、ローカルメモリ２９にアプリケーションｑを置くことは許される。そこで再生中断を承知の上、アプリケーションの読み出しを行う。つまり BD-ROM からローカルメモリ２９へとアプリケーションを読み出した上で、アプリケーションをワークメモリ３７に読み出す(ステップＳ１４０)。

ステップＳ１４１～ステップＳ１４６は、ステップＳ１３９が Yes と判定された場合になされる処理である。ステップＳ１４１では、読込属性を参照することで、アプリケーションがプリロードされているか否かを判定する。プリロードされていれば、ステップＳ１３５に移行する。

ステップＳ１４２は、読込属性がロードである場合に実行される判定ステップであり、アプリケーションｑがカルーセル化されているか、インターリーブ化されているかを判定する。インターリーブ化されていれば、キャッシュセンスを Java 仮想マシン３８に実行させる(ステップＳ１４３)。ローカルメモリ２

9にアプリケーションqが存在すれば、ステップS135に移行して、アプリケーションqをJava仮想マシン38にロードさせる。

ローカルメモリ29にアプリケーションがなければ、トップメニュータイトルに分岐する等の例外処理を行う(ステップS144)。カルーセル化されていれば、タイマをセットし(ステップS148)、そのタイマがタイムアウトするまで(ステップS147)、キャッシュセンスをJava仮想マシン38に実行させる(ステップS146)。もしローカルメモリ29にアプリケーションqが出現すれば、図49のステップS135に移行して、アプリケーションqをJava仮想マシン38にロードさせる。タイムアウトすれば、トップメニュータイトルに分岐する等の例外処理を行う(ステップS144)。

図51は、Java仮想マシン38によるアプリケーションの読み込みがどのようにして行われるかを模式化した図である。

矢印◎1,2は、アプリケーション管理テーブルに生存していて、データ管理テーブルに生存しており、カルーセル化,インターリーブ化を示す読込属性が存在するJavaアーカイブファイルの読み込みを示す。矢印◎1は、ステップS65、67においてなされるローカルメモリ29センスを示す。このローカルメモリ29センスは、カルーセル又はインターリーブ化により埋め込まれたデータが、ローカルメモリ29に存在するかもしれないためローカルメモリ29内をセンスするというものである。矢印◎2は、ステップS135に対応する読み込みであり、アプリケーションがローカルメモリ29に存在していた場合の、ローカルメモリ29からワークメモリ37へのロードを示す。×付きの矢印は、ローカルメモリ29にデータがない場合を示す。

矢印▽1,2は、アプリケーション管理テーブルに生存しているが、データ管理テーブルに生存しておらず、読込属性が存在しないJavaアーカイブファイルの読み込みを示す。

矢印▽1は、ステップS145における読み込みに対応するものであり、Java仮想マシン38によるBD-ROMからのダイレクトリードの要求を示す。矢印▽2はその要求による、BD-ROMからワークメモリ37へのJavaアーカイブファイル読み出しを示す。

矢印☆1,2,3は、アプリケーション管理テーブルに生存していて、データ管理テーブルに生存しているが、読込属性が存在しないJavaアーカイブファイルの読み込みを示す。

矢印☆1は、ステップS140における読み込みに対応するものであり、Java仮想マシン38によるBD-ROMからのダイレクトリードの要求を示す。矢印☆2はその要求による、ローカルメモリ29へのJavaアーカイブファイルの読み出しを示す。矢印☆3はローカルメモリ29からワークメモリ37へのJavaアーカイブファイルの読み出しを示す。

以上のように本実施形態によれば、ローカルメモリ29上で同時に常駐されるアプリケーションの数が所定数以下になるように規定しておくことができるので、ローカルメモリ29からの読み出し時におけるキャッシュミスを極力回避することができる。キャッシュミスのないアプリケーション読み出しを保証することができるので、アプリケーション呼出時にあたては、AVClipの再生を止めてまで、BD-ROMからアプリケーションを読み出すことはなくなる。AVClip再生を途切れさせないので、AVClipのシームレス再生を保証することができる。

(第7実施形態)

第3実施形態では、非AV系タイトルの時間軸をアプリケーションの生存区間に基づき定めることにした。しかしアプリケーションの動作というのは不安定であり、起動の失敗や異常終了がありうる。本実施形態は、起動失敗、異常終了があった場合のFail Safe機構を提案するものである。図52(a)は、第7実施形態に係るBD-Jオブジェクトの内部構成を示す図である。図7(b)と比較して本図が新規なのは、プレイリスト管理テーブルが追加されている点である。

図52(b)は、プレイリスト管理テーブルの一例を示す図である。本図に示すようにプレイリスト管理テーブルは、PLの指定と、そのPLの再生属性とからなる。PLの指定は、対応するタイトルのタイトル時間軸において、再生可能となるPLを示す。PLの再生属性は、指定されたPLを、タイトル再生の開始と同時に自動再生するか否かを示す(こうして自動再生されるPLをデフォ

ルト PL という)。

次にプレイリスト管理テーブルによりタイトル時間軸がどのように規定されるかを、図５３を参照しながら説明する。図５３（ａ）は、再生属性が非自動再生を示すよう設定された場合の非 AV 系タイトルにおけるタイトル時間軸を示す図である。この場合、デフォルト PL は再生されないから、非 AV 系タイトル同様、アプリケーションの生存区間からタイトル時間軸が定まる。

図５３（ｂ）は、再生属性が AutoPlay に設定された非 AV 系タイトルのタイトル時間軸を示す図である。再生属性が AutoPlay を示すよう設定されれば、Playback Control Engine３２は非 AV 系タイトルの再生開始と同時に、デフォルト PL の再生を開始する。しかしアプリケーションが正常に動作し、正常終了したとしても、このタイトル時間軸は、PL 時間軸を基準にして定められる。

図５３（ｃ）は、プレイリスト管理テーブルにおいて再生属性が" AutoPlay"を示すよう設定され、アプリケーションが異常終了した場合を示す。かかる異常終了により、どのアプリケーションも動作してない状態になるが、デフォルト PL の再生は継続する。この場合も、デフォルト PL の PL 時間軸がタイトル時間軸になる。

図５３（ｄ）は、プレイリスト管理テーブルにおいて再生属性が" AutoPlay"を示すよう設定され、メインアプリの起動に失敗したケースを示す。この場合も、Playback Control Engine３２によるデフォルト PL 再生は、アプリケーションの起動失敗とは関係なしに行われるので、デフォルト PL の時間軸がタイトル時間軸になる。

以上のようにプレイリスト管理テーブルの再生属性を、" AutoPlay" に設定しておけば、Java アプリケーションの起動に、5～10 秒という時間がかかったとしても、その起動がなされている間、" とりあえず何かが写っている状態" になる。この" とりあえず何かが写っている状態" によりタイトル実行開始時のスタートアップディレイを補うことができる。

以上は本実施形態における記録媒体に対する改良である。続いて本実施形態における再生装置に対する改良について説明する。

図５２（ｃ）は、分岐先タイトルのプレイリスト管理テーブルにおいて、再

生属性が AutoPlay に設定された PL が存在する場合、再生装置がどのような処理を行うかを示す図である。本図に示すように、再生属性が AutoPlay に設定された PL が、分岐先タイトルのプレイリスト管理テーブルに存在すれば、BD-J モジュール３５内のアプリケーションマネージャ３６は、タイトル分岐直後にこの AutoPlayPL の再生を開始するよう Playback Control Engine３２に指示する。このように再生属性が AutoPlay の PL は、タイトル分岐直後に再生開始が命じられることになる。

上述した記録媒体の改良に対応するため、アプリケーションマネージャ３６は図５４に示すような処理手順で処理を行う。

図５４は、第７実施形態に係るアプリケーションマネージャ３６の処理手順を示すフローチャートである。本フローチャートは、図３８のフローチャートにおいてステップＳ２１の前にステップＳ１０３、ステップＳ１０４を追加し、ステップＳ２１と、ステップＳ２２との間にステップＳ１００を追加し、ステップＳ２３－ステップＳ２６間に、ステップＳ１０５を追加したものである。

ステップＳ１０３は、対応するタイトルのプレイリスト管理テーブルの再生属性が AutoPlay であるか否かの判定である。もし AutoPlay なら、デフォルト PL に対する再生制御を Playback Control Engine３２に開始させる(ステップＳ１０４)。

ステップＳ１００は、Presentation Engine３１による再生中であるか否かを判定する。もし再生中であるなら、ステップＳ１０１に移行する。

ステップＳ１０５は、ステップＳ２３が Yes、ステップＳ２５が No である場合に実行される判定ステップであり、再生属性が AutoPlay であるか否かを示す。もし否であるなら、タイトル終了をモジュールマネージャ３４に通知する。もし AutoPlay であるなら、ステップＳ１０１に移行して、処理を継続する。

図５５は、プレイリスト管理テーブルにおいて”再生属性＝AutoPlay”に設定されることにより、どのような再生が行われるかを模式化した図である。ここで再生すべきタイトルは、落下するタイル片を積み重ねるというゲームアプリを含む非 AV 系タイトルである。この非 AV 系タイトルにおいて、プレイリ

スト管理テーブルの再生属性が AutoPlay に設定されていれば、Playback Control Engine３２によるデフォルト PL 再生も開始する。ゲームアプリの実行と、デフォルト PL 再生とが並列的になされるので、図５５の上段の左側に示すように、前景をゲームアプリの画面とし、背景をデフォルト PL の再生画像とした合成画像が表示されることになる。このゲームアプリは途中で異常終了したとする。ゲームアプリはアプリケーションマネージャ３６により強制終了させられるが、デフォルト PL の再生が継続してなされるため、タイトルは、何かが写っている状態になる。このようなプレイリスト管理テーブルにおける再生属性の指定により、非 AV 系タイトル内のゲームアプリが異常終了した場合でも、ハングアップやブラックアウトがない動作を維持することができる。

(第８実施形態)

第１実施形態において BD-J オブジェクトは、データ管理テーブル、アプリケーション管理テーブルという２つのテーブルを具備していたが、本実施形態は、これらを１つのテーブルに統合するという形態を開示する。かかる統合にあたって、図５６（ａ）に示すように、データ管理テーブルにおける読込属性という項目を廃し、代わりに起動属性に Ready 属性という属性を設ける。Ready 属性とは、他のアプリケーションからの呼出又はアプリケーションマネージャ３６からの呼出に備えて、ローカルメモリ２９に予めアプリケーションをロードしておく旨を示す起動属性の類型である。

図５６（ｂ）は、アプリケーションの扱いと、起動属性との関係を示した図である。第１実施形態に示したようにアプリケーションの扱いには、プリロードされるか否か(1)、現在の再生時点が有効区間に到来した際自動的に起動されるか、他からの呼出に応じて起動されるか(2)、タイトル再生進行に従ってロードされるか(3)、生存しているかという違いがあり、これらの違いにより、図５６（ｂ）に示すような 5 つの態様が出現する。このうち起動属性が AutoRun に設定されるのは、プリロードがなされ、”自動起動”である場合、及び、ロードがなされ、”自動起動”である場合である。

一方、起動属性が Ready 属性に設定されるのは、プリロード、又は、ロードがなされ、起動項目が”呼出起動”を示している場合である。

尚、ワークメモリ３７では生存しているが、ローカルメモリ２９にはロードされない”との類型が存在し得ない。これは、アプリケーション・データ管理テーブルでは、ワークメモリ３７の生存区間と、ローカルメモリ２９の生存区間とが一体だからである。

起動属性として、この Ready 属性を追加されたので、アプリケーションマネージャ３６はタイトル再生に先立ち、起動属性が AutoRun に設定されたアプリケーション、及び、起動属性が Ready 属性に設定されたアプリケーションをローカルメモリ２９にプリロードするとの処理を行う。こうすることにより、読込属性を設けなくても、アプリケーションをローカルメモリ２９にプリロードしておくとの処理が可能になる。

図５７は、第８実施形態に係る Java 仮想マシン３８によるアプリケーションの読み込みがどのようにして行われるかを模式化した図である。本図における読み込みは、図５１をベースにして作図している。

矢印◎1,2 は、アプリケーション・データ管理テーブルに生存していて、起動属性が Ready 属性に設定されている Java アーカイブファイルの読み込みを示す。

矢印☆1,2,3 は、アプリケーション・データ管理テーブルに生存していおり、起動属性が Persistent であるアプリケーションの読み込みを示す。

これらの矢印◎1,2、矢印☆1,2,3 は、図５１でも記述されていたものだが、図５１に記述していた、▽1,2 の矢印に該当する読み込み”は、図５７では存在しない。これは、アプリケーション・データ管理テーブルは、アプリケーション管理テーブル、データ管理テーブルを一体化したものなので、アプリケーション管理テーブル＝生存、データ管理テーブル＝非存在という組合せは表現し得ないからである。

以上のように本実施形態によれば、データ管理テーブル、アプリケーション管理テーブルを１つのテーブル(アプリケーション・データ管理テーブル)にまとめることができるので、アプリケーションマネージャ３６による処理を簡略化することができる。尚、読込優先度をなくすことによりアプリケーション・データ管理テーブルをより簡略化にしても良い。

(第９実施形態)

第１実施形態では、アプリケーションをローカルメモリ２９に読み込むにあたって、読込優先度を参照して、この読込優先度に従い、読み込み処理に優劣を与えた。これに対し第９実施形態は、Optional を意味する情報と、0 から 255 までの数値との組合せにより読込優先度を表す実施形態である。

図５８（ａ）（ｂ）は、第９実施形態に係る読込優先度の一例を示す図である。255、128 は、0 から 255 までの読込優先度の一例であり、本例における application#2 は、application#3 より読込優先度が高いことを意味する。

本実施形態においてアプリケーションマネージャ３６は、第１実施形態同様、先ず Mandatory を示す読込優先度が付与されたアプリケーションをローカルメモリ２９に読み込む。

その後、Optional を示す読込優先度が付与されたアプリケーションに対しては、ローカルメモリ２９における容量が、アプリケーションのサイズを上回るか否かを判定する。もし上回るなら、読込優先度＝Optional が付与されたアプリケーションをそのままローカルメモリ２９に読み込む。もし下回るなら、アプリケーションを構成するデータのうち、読込優先度を表す数値が高いアプリケーションをローカルメモリ２９に読み込む。そして、ローカルメモリ２９における残りの領域に、読込優先度を表す数値が低いアプリケーションを読み出す。

こうすることで Optional 扱いのアプリケーションについては、全体を格納する容量が再生装置のローカルメモリ２９になくても、その一部分をローカルメモリ２９に格納しておくことができる。

(第１０実施形態)

第１実施形態においてアプリケーションマネージャ３６は、同じ applicationID が付与されたアプリケーションを、読込優先度に従い排他的にローカルメモリ２９にロードするとしたが、第１０実施形態は、アプリケーションにグループ属性を与えることにより、排他的なロードを実現する。図５９は、グループ属性が付与されたデータ管理テーブルを示す図である。グループ

属性には、排他グループなし、排他グループあり、といった、2 通りの設定が可能であり、排他グループありの場合、そのグループ番号が記述される。図５９（ａ）における title#1 の「－」は、排他グループが存在しないことを示す。一方、title#2,#3 の「group#1」は、排他グループがあり、title#2,#3 は、group#1 という排他グループに帰属していることを示す。以上が本実施形態に係る記録媒体の改良である。

本実施形態に係る再生装置は、データ管理テーブルに基づいて各アプリケーションをローカルメモリ２９に読み込んだ後、ローカルメモリ２９のアプリケーションにおけるグループ属性をベリファイする。同じ排他グループに帰属するアプリケーションが、ローカルメモリ２９上に２つ以上存在していれば、そのうち一方をローカルメモリ２９から削除する。

こうすることにより、ローカルメモリ２９の利用効率を向上させることができる。排他グループの具体例としては、ランチャーアプリと、このアプリにより起動されるアプリとからなるグループが相応しい。本アプリケーションにより起動されるアプリケーションは、原則１つに限られるので、ローカルメモリ２９には、ランチャー＋１個のアプリケーションのみが存在する筈である。もし３つ以上のアプリケーションが存在していれば、これをローカルメモリ２９から削除するという処理をアプリケーションマネージャ３６は行う必要があるので、各アプリケーションのグループ属性を設け、ローカルメモリ２９上で存在するアプリケーションがランチャー＋１個のアプリケーションになっているかどうかのチェックを行うのである。

図５９（ａ）は、アプリケーション管理テーブルに基づくローカルメモリ２９に対するアクセスを示す図である。本図において、読込優先度＝Optional と設定された application#2、application#3 のグループ属性は、group#1 であるので、これらのアプリケーションは、同じ排他グループに属することになる。３つのアプリケーションのうち、application#1 は上述したランチャーアプリケーションであり、application#2、application#3 は、これにより起動されるアプリケーションであるので、どちらかのみがローカルメモリ２９上に存在するよう、グループ属性が付与されている。アプリケーションマネージャ３６は、

これら application#2、application#3 のグループ属性を参照して、どちらか１つをローカルメモリ２９から削除するとの処理を行う。かかる削除によりローカルメモリ２９に余白が生まれる。

(第１１実施形態)

第１実施形態では、アプリケーション管理テーブルをタイトル毎に持たせるとしたが、本実施形態では、このアプリケーション管理テーブルの割当単位を変更させることを提案する。図６０は、割当単位のバリエーションを示す図である。本図において第１段目は、BD-ROM に記録されている３つのアプリケーション管理テーブルを示し、第２段目は、タイトル単位、第３段目は、ディスク単位、第４段目は、複数 BD-ROM からなるディスクセット単位を示す。図中の矢印は、アプリケーション管理テーブルの割り当てを模式化して示している。この矢印を参照すると、第１段目におけるアプリケーション管理テーブル#1,#2,#3 のそれぞれは、第２段目に示した title#1,#2,#3 のそれぞれに割り当てられていることがわかる。また、ディスク単位ではアプリケーション管理テーブル#4 が割り当てられており、ディスクセット全体に対しはアプリケーション管理テーブル#5 が割り当てられている。このようにアプリケーション管理テーブルの割当単位を、タイトルより大きい単位にすることにより、１つの BD-ROM がローディングされている間、生存するようなアプリケーションや複数 BD-ROM のうちどれかがローディングされている間、生存するようなアプリケーションを定義することができる。

(備考)

以上の説明は、本発明の全ての実施行為の形態を示している訳ではない。下記(A)(B)(C)(D)・・・・・の変更を施した実施行為の形態によっても、本発明の実施は可能となる。本願の請求項に係る各発明は、以上に記載した複数の実施形態及びそれらの変形形態を拡張した記載、ないし、一般化した記載としている。拡張ないし一般化の程度は、本発明の技術分野の、出願当時の技術水準の特性に基づく。

(A)全ての実施形態では、本発明に係る光ディスクを BD-ROM として実施し

たが、本発明の光ディスクは、記録される動的シナリオ、Index Table に特徴があり、この特徴は、BD-ROM の物理的性質に依存するものではない。動的シナリオ、Index Table を記録しうる記録媒体なら、どのような記録媒体であってもよい。例えば、

DVD-ROM,DVD-RAM,DVD-RW,DVD-R,DVD+RW,DVD+R,CD-R,CD-RW 等の光ディスク、PD,MO 等の光磁気ディスクであってもよい。また、コンパクトフラッシュカード、スマートメディア、メモリスティック、マルチメディアカード、PCM-CIA カード等の半導体メモリカードであってもよい。フレシキブルディスク、SuperDisk,Zip,Clik! 等の磁気記録ディスク(i)、ORB,Jaz,SparQ,SyJet,EZFley,マイクロドライブ等のリムーバルハードディスクドライブ(ii)であってもよい。更に、機器内蔵型のハードディスクであってもよい。

(B) 全ての実施形態における再生装置は、BD-ROM に記録された AVClip をデコードした上で TV に出力していたが、再生装置を BD-ROM ドライブのみとし、これ以外の構成要素を TV に具備させてもい、この場合、再生装置と、TV とを IEEE1394 で接続されたホームネットワークに組み入れることができる。また、実施形態における再生装置は、テレビと接続して利用されるタイプであったが、ディスプレィと一体型となった再生装置であってもよい。更に、各実施形態の再生装置において、処理の本質的部分をなす部分のみを、再生装置としてもよい。これらの再生装置は、何れも本願明細書に記載された発明であるから、これらの何れの態様であろうとも、各実施形態に示した再生装置の内部構成を元に、再生装置を製造する行為は、本願の明細書に記載された発明の実施行為になる。各実施形態に示した再生装置の有償・無償による譲渡(有償の場合は販売、無償の場合は贈与になる)、貸与、輸入する行為も、本発明の実施行為である。店頭展示、カタログ勧誘、パンフレット配布により、これらの譲渡や貸渡を、一般ユーザに申し出る行為も本再生装置の実施行為である。

(C)各フローチャートに示したプログラムによる情報処理は、ハードウェア資源を用いて具体的に実現されていることから、上記フローチャートに処理手順

を示したプログラムは、単体で発明として成立する。全ての実施形態は、再生装置に組み込まれた態様で、本発明に係るプログラムの実施行為についての実施形態を示したが、再生装置から分離して、各実施形態に示したプログラム単体を実施してもよい。プログラム単体の実施行為には、これらのプログラムを生産する行為(1)や、有償・無償によりプログラムを譲渡する行為(2)、貸与する行為(3)、輸入する行為(4)、双方向の電子通信回線を介して公衆に提供する行為(5)、店頭展示、カタログ勧誘、パンフレット配布により、プログラムの譲渡や貸渡を、一般ユーザに申し出る行為(6)がある。

(D)各フローチャートにおいて時系列に実行される各ステップの「時」の要素を、発明を特定するための必須の事項と考える。そうすると、これらのフローチャートによる処理手順は、再生方法の使用形態を開示していることがわかる。各ステップの処理を、時系列に行うことで、本発明の本来の目的を達成し、作用及び効果を奏するよう、これらのフローチャートの処理を行うのであれば、本発明に係る記録方法の実施行為に該当することはいうまでもない。

(E)Chapter を一覧表示するための Menu(Chapter Menu)と、これの挙動を制御する MOVIE オブジェクトとを BD-ROM に記録しておき、Top Menu から分岐できるようにしてもよい。またリモコンキーの Chapter キーの押下により呼出されるようにしてもよい。

(F)BD-ROM に記録するにあたって、AVClip を構成する各 TS パケットには、拡張ヘッダを付与しておくことが望ましい。拡張ヘッダは、TP_extra_header と呼ばれ、『Arribval_Time_Stamp』と、『copy_permission_indicator』とを含み 4 バイトのデータ長を有する。TP_extra_header 付き TS パケット(以下 EX 付き TS パケットと略す)は、32 個毎にグループ化されて、3 つのセクタに書き込まれる。32 個の EX 付き TS パケットからなるグループは、6144 バイト(=32×192)であり、これは 3 個のセクタサイズ 6144 バイト(=2048×3)と一致する。3 個のセクタに収められた 32 個の EX 付き TS パケットを" Aligned Unit" という。

IEEE1394 を介して接続されたホームネットワークでの利用時において、再生装置２００は、以下のような送信処理にて Aligned Unit の送信を行う。つま

り送り手側の機器は、Aligned Unit に含まれる 32 個の EX 付き TS パケットのそれぞれから TP_extra_header を取り外し、TS パケット本体を DTCP 規格に基づき暗号化して出力する。TS パケットの出力にあたっては、TS パケット間の随所に、isochronous パケットを挿入する。この挿入箇所は、TP_extra_header の Arribval_Time_Stamp に示される時刻に基づいた位置である。TS パケットの出力に伴い、再生装置２００は DTCP_Descriptor を出力する。DTCP_Descriptor は、TP_extra_header におけるコピー許否設定を示す。ここで「コピー禁止」を示すよう DTCP_Descriptor を記述しておけば、IEEE1394 を介して接続されたホームネットワークでの利用時において TS パケットは、他の機器に記録されることはない。

(G)各実施形態において、記録媒体に記録されるデジタルストリームは AVClip であったが、DVD-Video 規格、DVD-Video Recording 規格の VOB(Video Object)であってもよい。VOB は、ビデオストリーム、オーディオストリームを多重化することにより得られた ISO/IEC13818-1 規格準拠のプログラムストリームである。また AVClip におけるビデオストリームは、MPEG4 や WMV 方式であってもよい。更にオーディオストリームは、Linear-PCM 方式、Dolby-AC3 方式、MP3 方式、MPEG-AAC 方式、Dts、WMA(Windows media audio)であってもよい。

(H)各実施形態における映像作品は、アナログ放送で放送されたアナログ映像信号をエンコードすることにより得られたものでもよい。デジタル放送で放送されたトランスポートストリームから構成されるストリームデータであってもよい。

またビデオテープに記録されているアナログ／デジタルの映像信号をエンコードしてコンテンツを得ても良い。更にビデオカメラから直接取り込んだアナログ／デジタルの映像信号をエンコードしてコンテンツを得ても良い。他にも、配信サーバにより配信されるデジタル著作物でもよい。

(I)BD-J モジュール３５は、衛星放送受信のために機器に組み込まれた Java プラットフォームであってもよい。BD-J モジュール３５がかかる Java プラットフォームであれば、本発明に係る再生装置は、MHP 用 STB としての処理を

兼用することになる。

更に携帯電話の処理制御のために機器に組み込まれた Java プラットフォームであってもよい。かかる BD-J モジュール３５がかかる Java プラットフォームであれば、本発明に係る再生装置は、携帯電話としての処理を兼用することになる。

(K)レイアモデルにおいて、BD-J モードの上に MOVIE モードを配置してもよい。特に MOVIE モードでの動的シナリオの解釈や、動的シナリオに基づく制御手順の実行は、再生装置に対する負担が軽いので、MOVIE モードを BD-J モード上で実行させても何等問題は生じないからである。また再生装置や映画作品の開発にあたって、動作保証が１つのモードで済むからである。

更に BD-J モードだけで再生処理を実行してもよい。第５実施形態に示したように、BD-J モードでも PL の再生と同期した再生制御が可能になるから、強いて MOVIE モードを設けなくてもよいという理由による。

(L)AVClip に多重化されるべきインタラクティブグラフィクスストリームにナビゲーションコマンドを設けて、ある PL から別の PL への分岐を実現しても良い。

産業上の利用可能性

本発明に係る再生装置は、ホームシアターシステムでの利用のように、個人的な用途で利用されることがありうる。しかし本発明は上記実施形態に内部構成が開示されており、この内部構成に基づき量産することが明らかなので、資質において工業上利用することができる。このことから本発明に係る再生装置は、産業上の利用可能性を有する。

請求の範囲

1．タイトルの再生と、アプリケーションの実行とを行う再生装置であって、
タイトルに帰属するデジタルストリームを再生する再生制御エンジン部と、
複数タイトル間の分岐を制御するモジュールマネージャと、
1つ以上のアプリケーションを実行するモジュールとを備え、
前記モジュールは、仮想マシン部と、アプリケーションマネージャを含み、
前記アプリケーションマネージャは、
1つ以上のアプリケーションが所定の状態になった際、タイトル実行が終了したとして解釈して終了処理を行い、
前記モジュールマネージャは、前記タイトルの終了後、所定のタイトルを選択する、ことを特徴とする再生装置。

2．アプリケーションにおける所定の状態とは、
1つ以上のアプリケーションが全て終了した状態である
ことを特徴とする請求項1記載の再生装置。

3．1つ以上のアプリケーションには、メインアプリケーションがあり、
メインアプリケーションは、タイトル開始と同時に自動的に起動される唯一のアプリケーションであり、
アプリケーションにおける所定の状態とは、
アプリケーションのうち、メインアプリケーションが終了した状態である、
請求項1記載の再生装置。

4．モジュールマネージャにより選択される所定のタイトルとは、トップメニューの表示制御を実行するトップメニュータイトルである、請求項2又は3記載の再生装置。

5．前記記録媒体には、タイトル開始時において再生を自動的に開始すべき

再生経路を規定する規定情報が記述されており、

前記モジュールは、タイトル開始時において、アプリケーションの起動を仮想マシンに指示すると共に、前記規定情報に示される再生経路に従い、デジタルストリームの再生を開始するよう再生制御エンジン部に指示し、

再生制御エンジン部は、

仮想マシンによるアプリケーション実行が終了した際、所定の再生経路に従ったデジタルストリームの再生出力を継続して行う、請求項1記載の再生装置。

6．タイトルの再生と、アプリケーションの実行とを同時にコンピュータに実行させるプログラムであって、

1つ以上のアプリケーションが所定の状態になった際、タイトル実行が終了したとして解釈してアプリケーションの終了処理を行い、前記タイトルの終了後、所定のタイトルを選択するようコンピュータを制御する、ことを特徴とするプログラム。

7．タイトルの再生と、アプリケーションの実行とを同時にコンピュータに実行させる再生方法であって、

1つ以上のアプリケーションが所定の状態になった際、タイトル実行が終了したとして解釈してアプリケーションの終了処理を行い、前記タイトルの終了後、所定のタイトルを選択するようコンピュータを制御する、ことを特徴とする再生方法。

図1

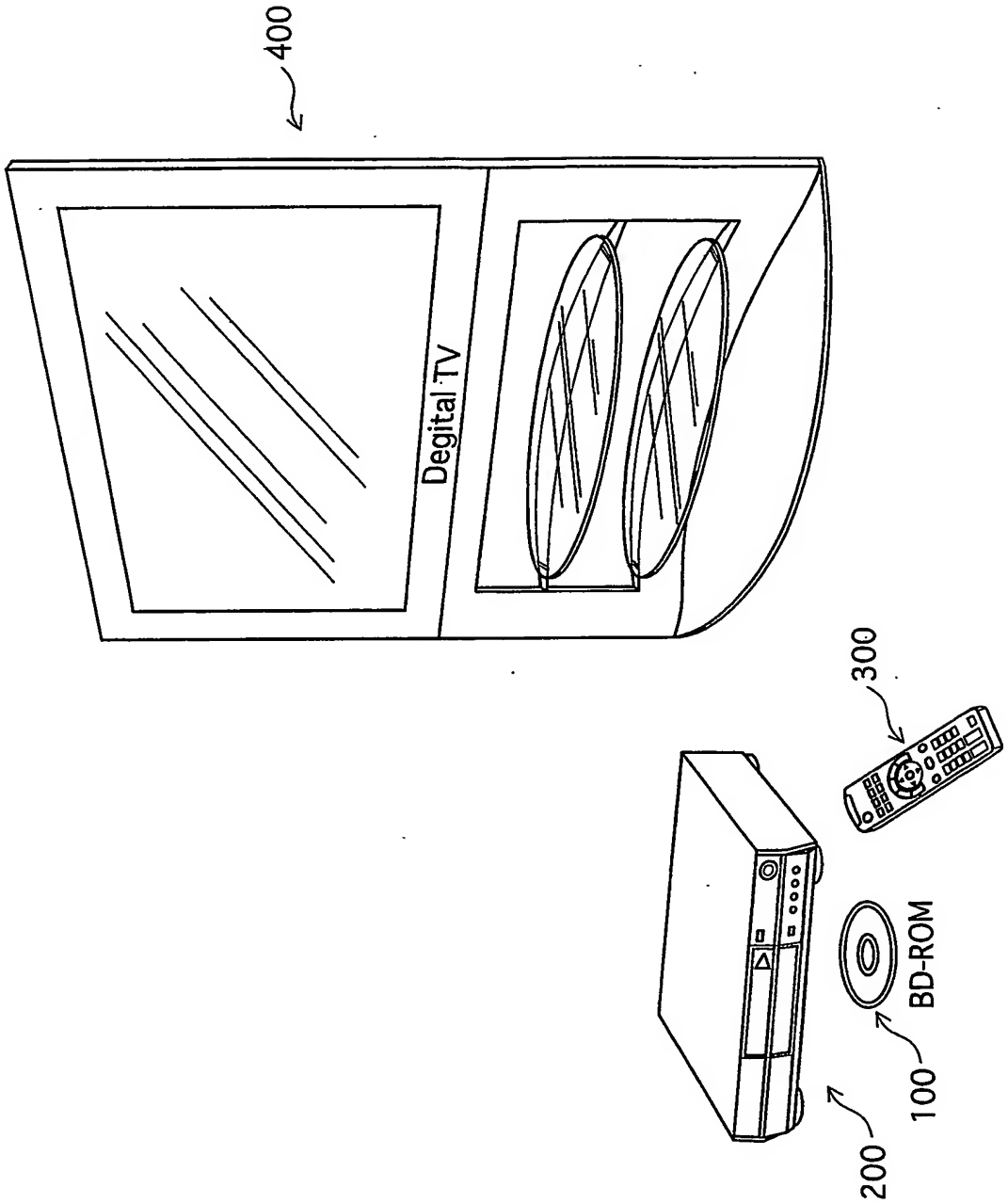


図2

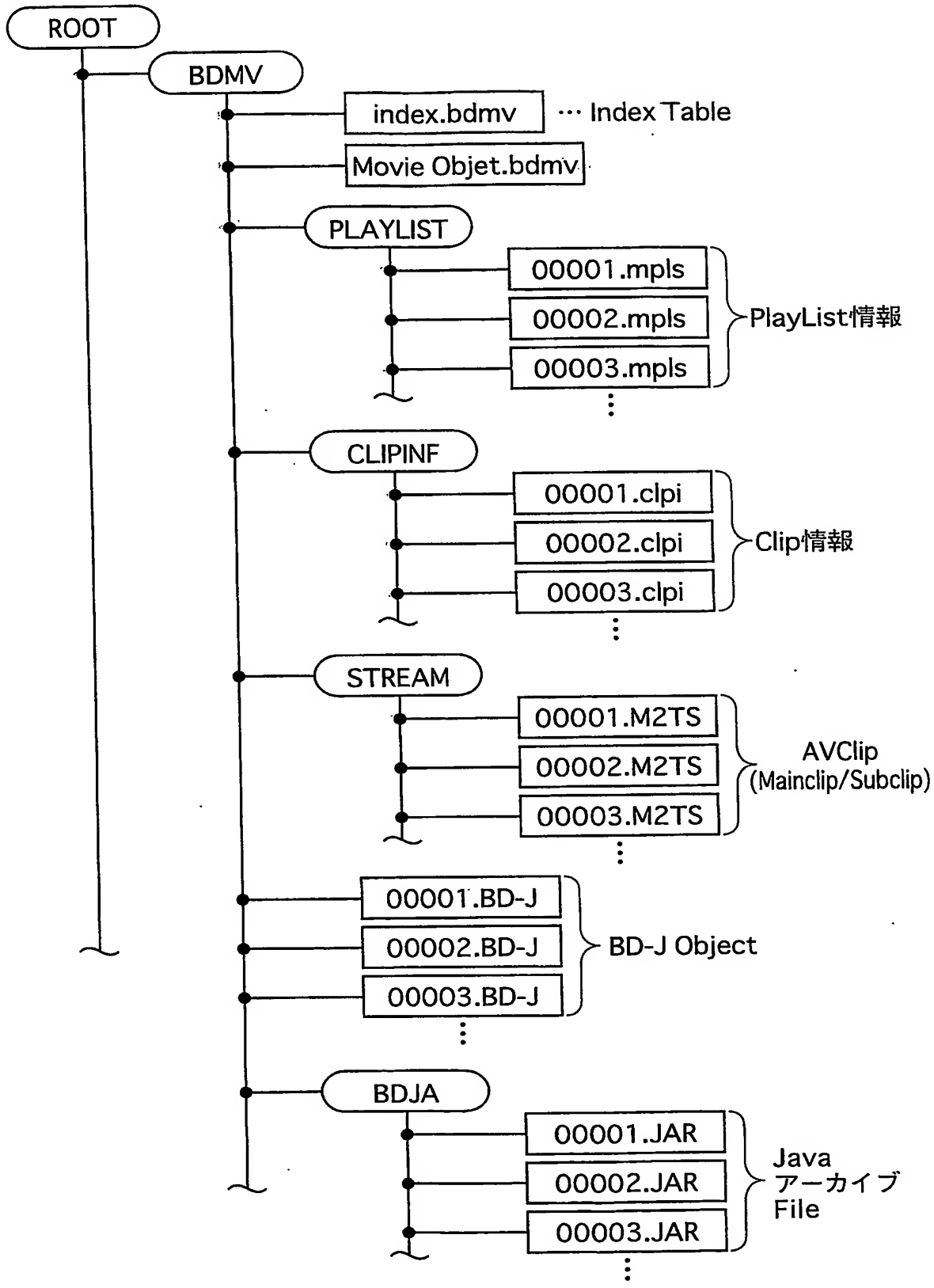

ROOT
BDMV
index.bdmv
… Index Table
Movie Objet.bdmv
PLAYLIST
00001.mpls
00002.mpls
00003.mpls
PlayList情報
CLIPINF
00001.clpi
00002.clpi
00003.clpi
Clip情報
STREAM
00001.M2TS
00002.M2TS
00003.M2TS
AVClip
(Mainclip/Subclip)
00001.BD-J
00002.BD-J
00003.BD-J
BD-J Object
BDJA
00001.JAR
00002.JAR
00003.JAR
Java
アーカイブ
File

図3

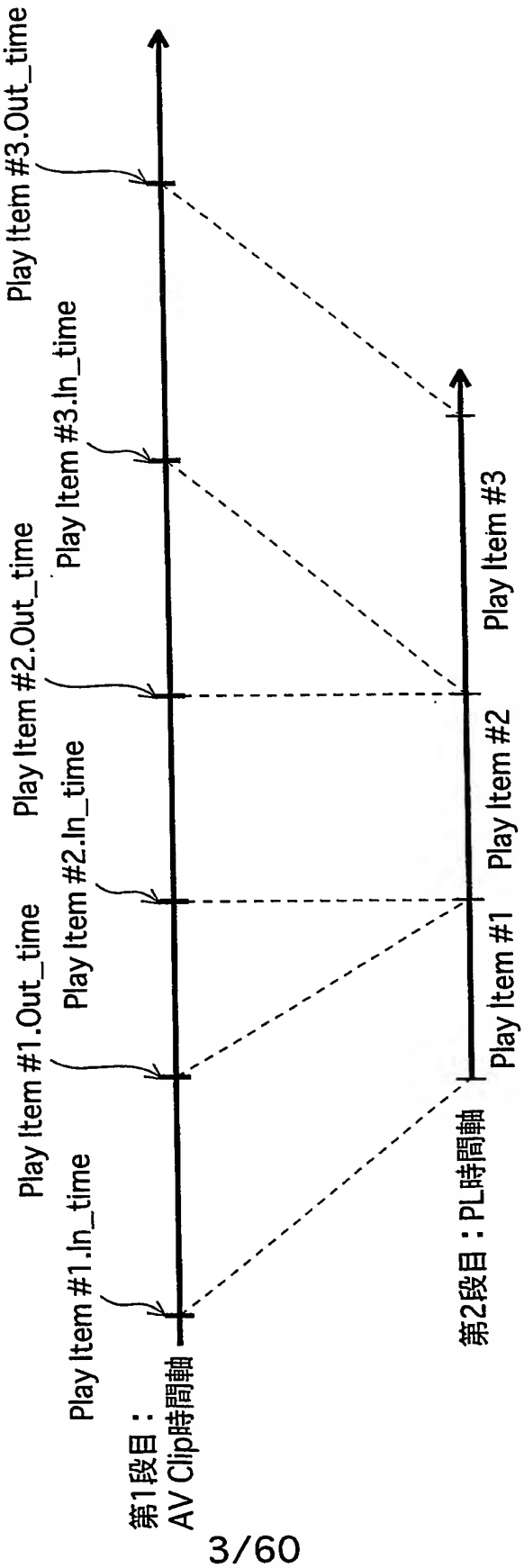

Play Item #3.Out_time
Play Item #3.In_time
Play Item #2.Out_time
Play Item #3
Play Item #2.In_time
Play Item #2
Play Item #1.Out_time
Play Item #1
Play Item #1.In_time
第1段目：
AV Clip時間軸
第2段目：PL時間軸

図4

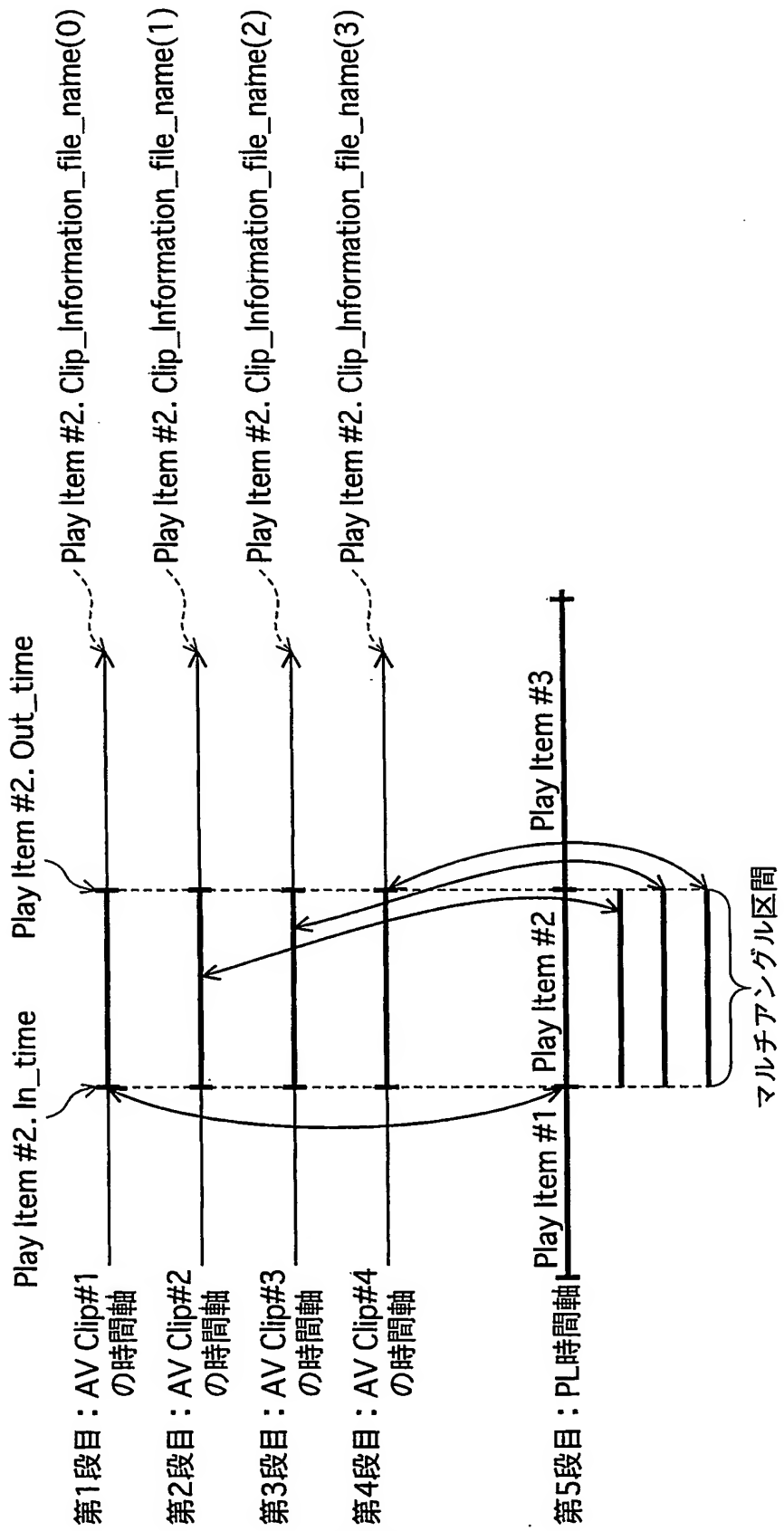

Play Item #2. Clip_Information_file_name(0)
Play Item #2. Clip_Information_file_name(1)
Play Item #2. Clip_Information_file_name(2)
Play Item #2. Clip_Information_file_name(3)
Play Item #2. Out_time
Play Item #2. In_time
Play Item #3
Play Item #2
Play Item #1
第1段目：AV Clip#1の時間軸
第2段目：AV Clip#2の時間軸
第3段目：AV Clip#3の時間軸
第4段目：AV Clip#4の時間軸
第5段目：PL時間軸
マルチアングル区間

図5

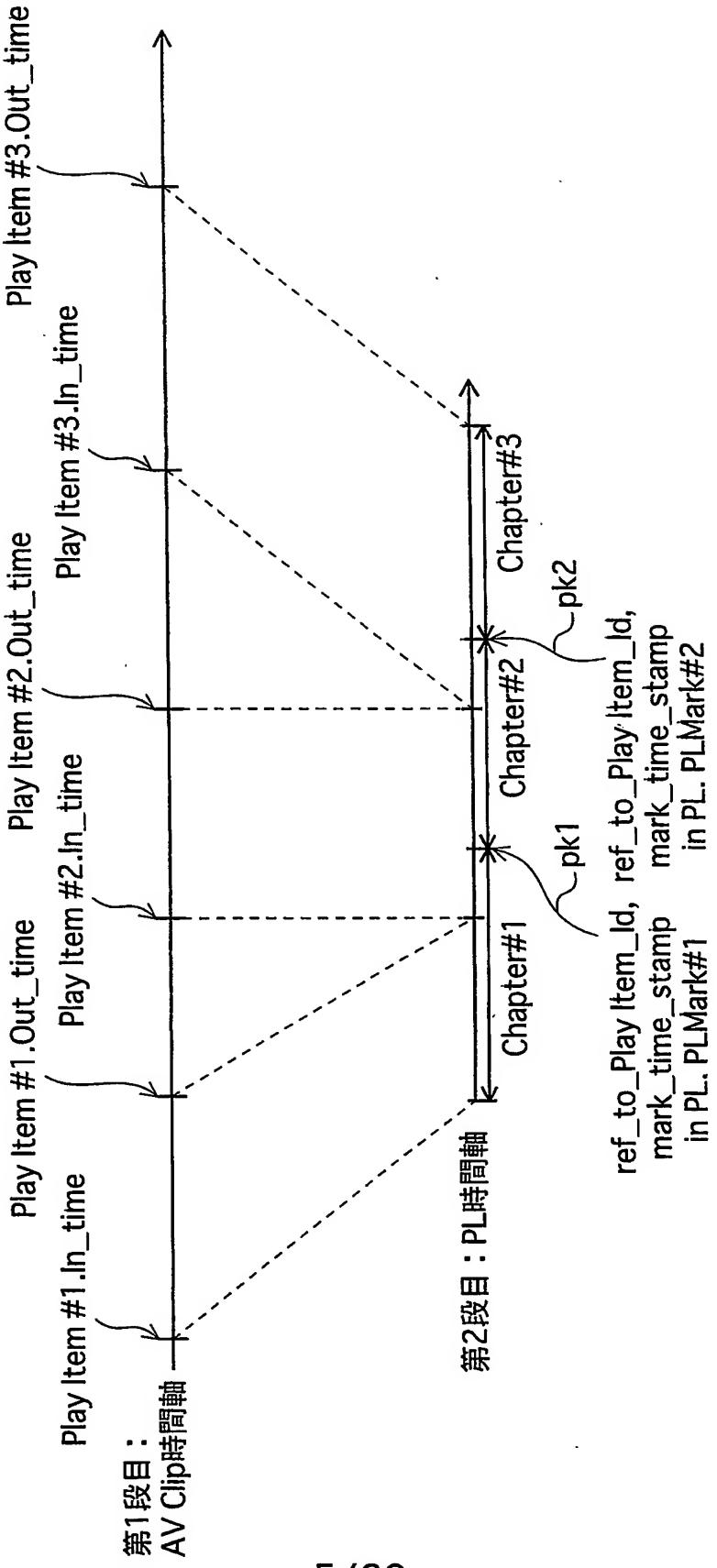

第1段目：
AV Clip時間軸
第2段目：PL時間軸
Play Item #1.In_time
Play Item #1.Out_time
Play Item #2.In_time
Play Item #2.Out_time
Play Item #3.In_time
Play Item #3.Out_time
Chapter#1
Chapter#2
Chapter#3
pk1
pk2
ref_to_Play Item_Id,
mark_time_stamp
in PL. PLMark#1
ref_to_Play Item_Id,
mark_time_stamp
in PL. PLMark#2

図6

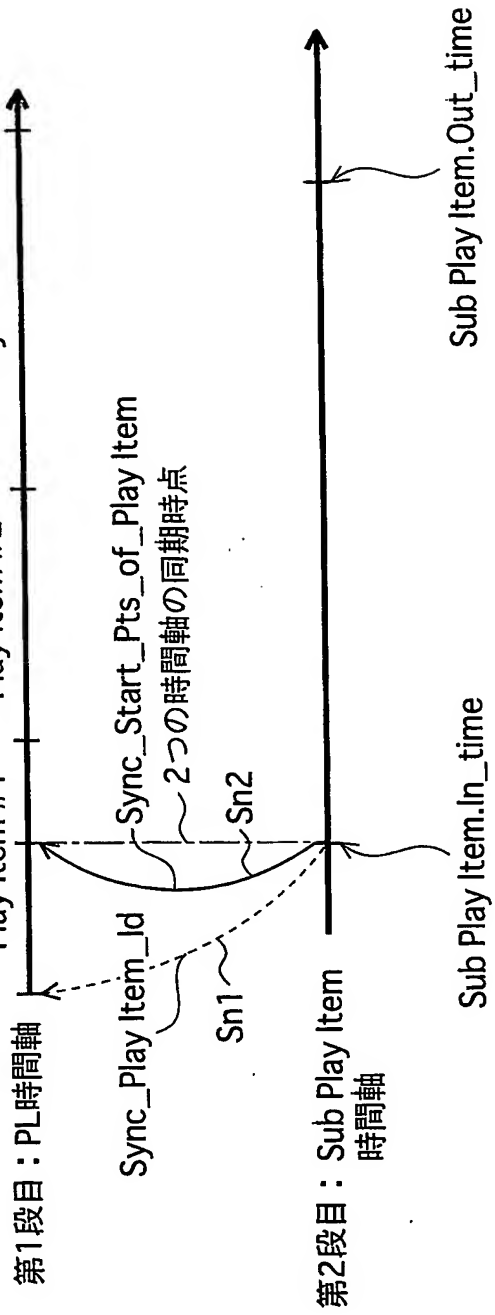
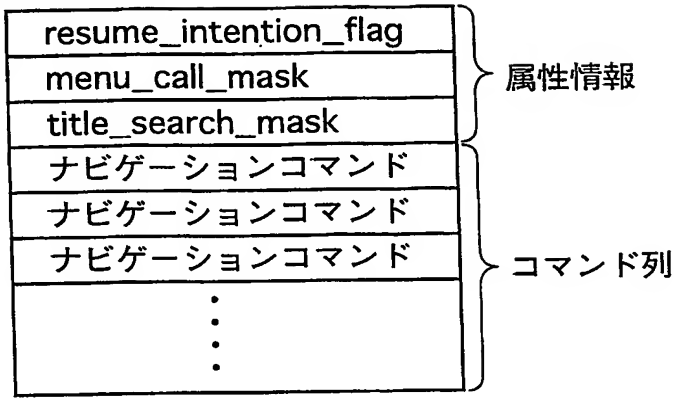

Play Item #1
Play Item #2
Play Item #3
Sync_Start_Pts_of_Play Item
2つの時間軸の同期時点
Sn2
Sn1
Sync_Play Item_Id
Sub Play Item.In_time
Sub Play Item.Out_time
第1段目：PL時間軸
第2段目： Sub Play Item
時間軸

図7

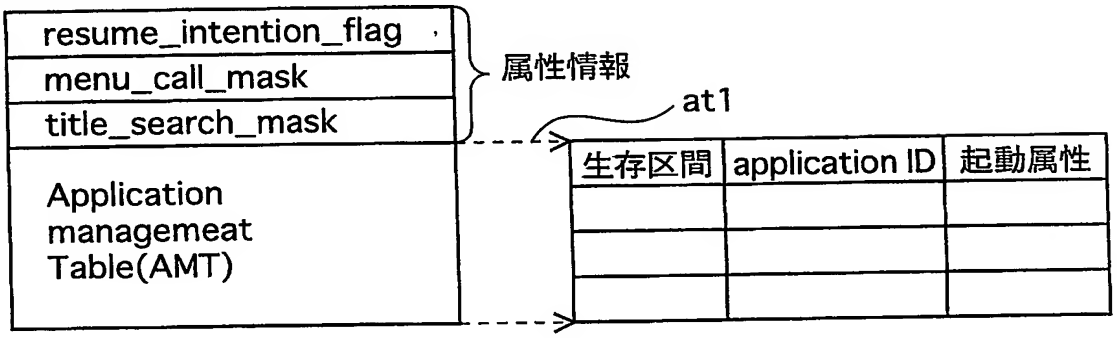
(a)

ZZZZZ.BDMV



(b)

ZZZZZ.BD-J



(c)

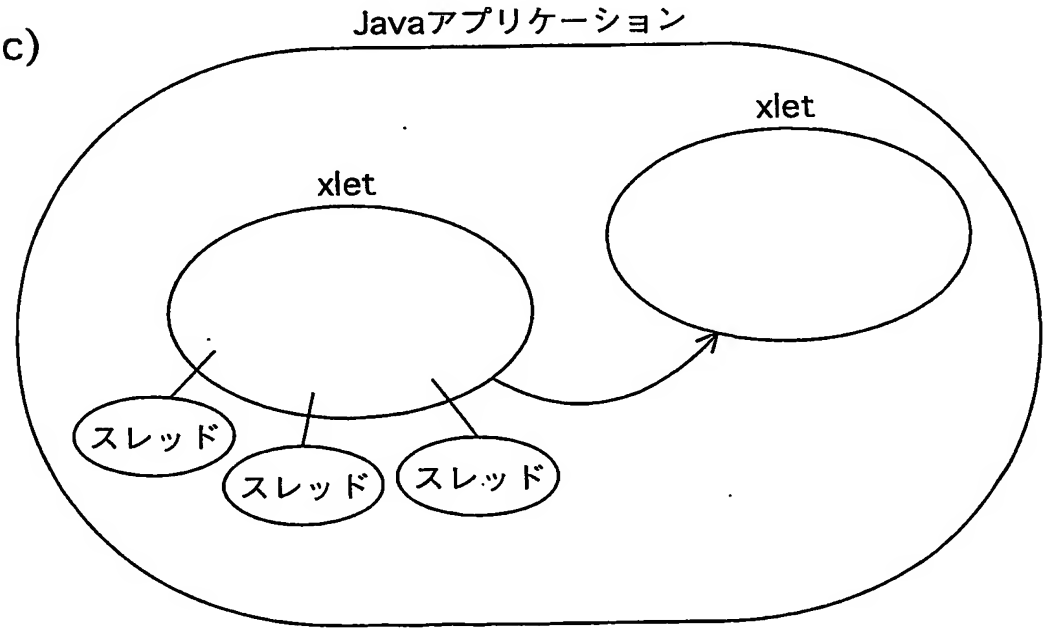


図8

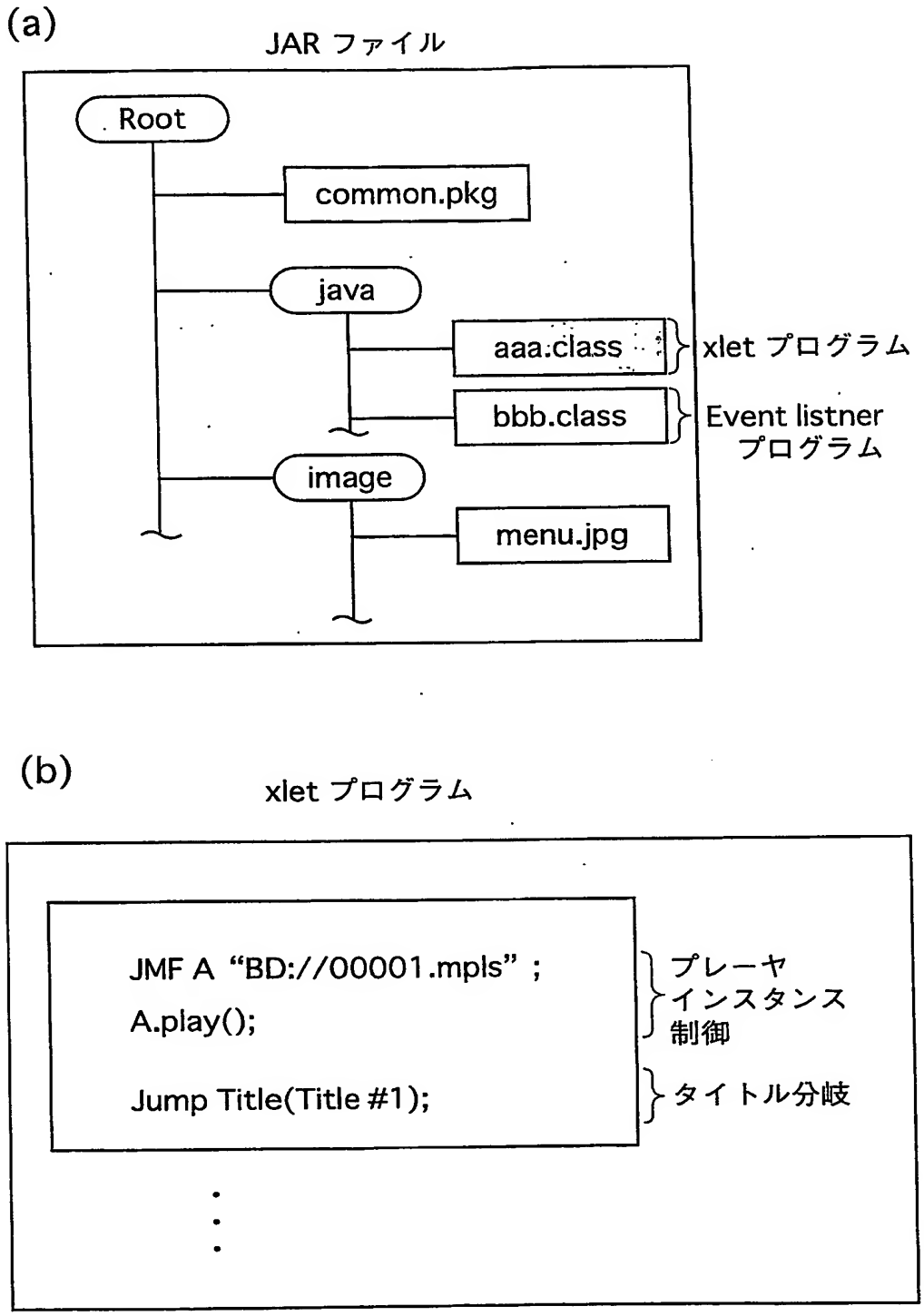

(a)
JAR ファイル
Root
common.pkg
java
aaa.class
xlet プログラム
bbb.class
Event listner
プログラム
image
menu.jpg
(b)
xlet プログラム
JMF A "BD://00001.mpls" ;
A.play();
プレーヤ
インスタンス
制御
Jump Title(Title #1);
タイトル分岐

図9

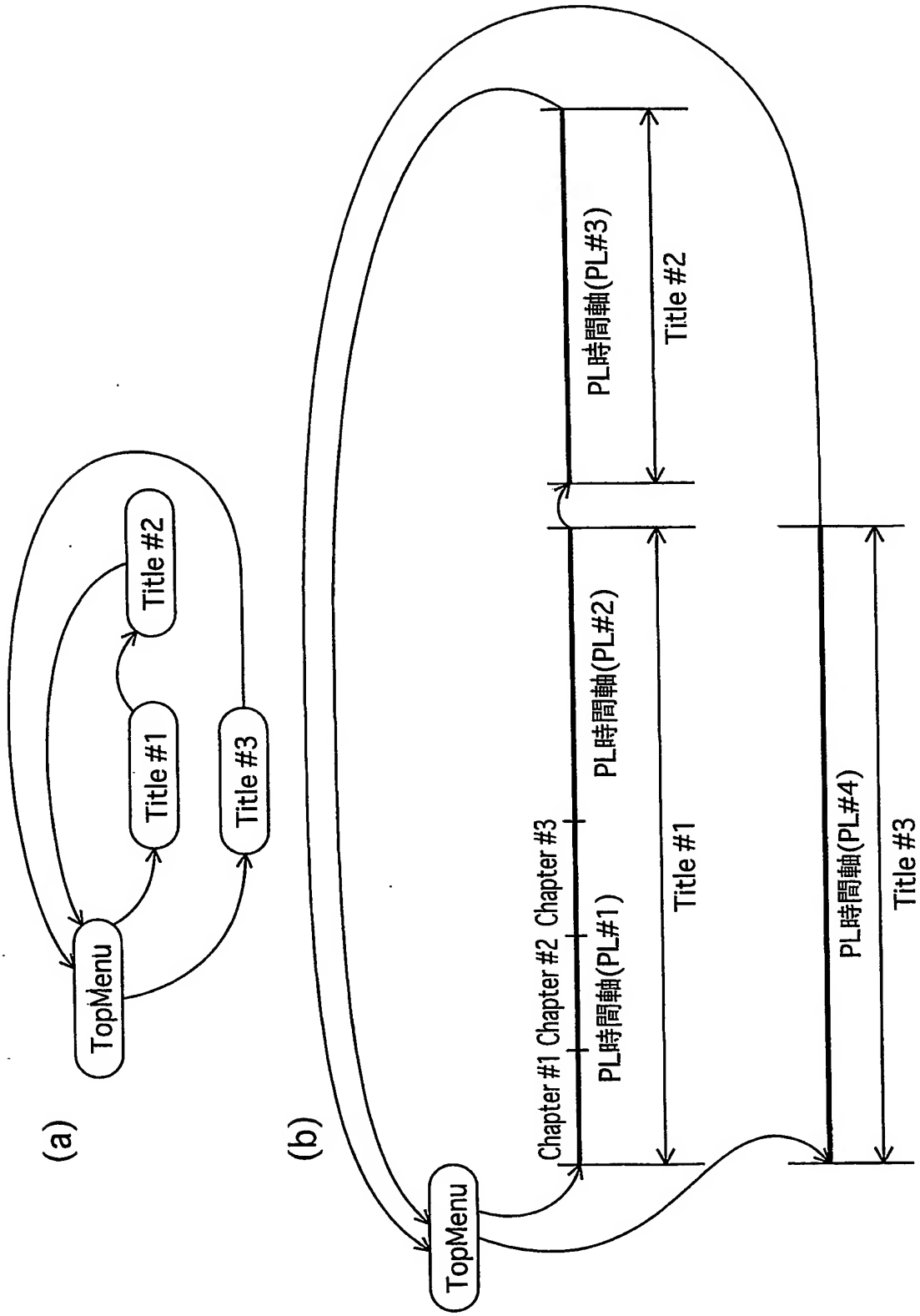

(a)
TopMenu
Title #1
Title #2
Title #3
(b)
TopMenu
Chapter #1
Chapter #2
Chapter #3
PL時間軸(PL#1)
PL時間軸(PL#2)
PL時間軸(PL#3)
PL時間軸(PL#4)
Title #1
Title #2
Title #3

図10

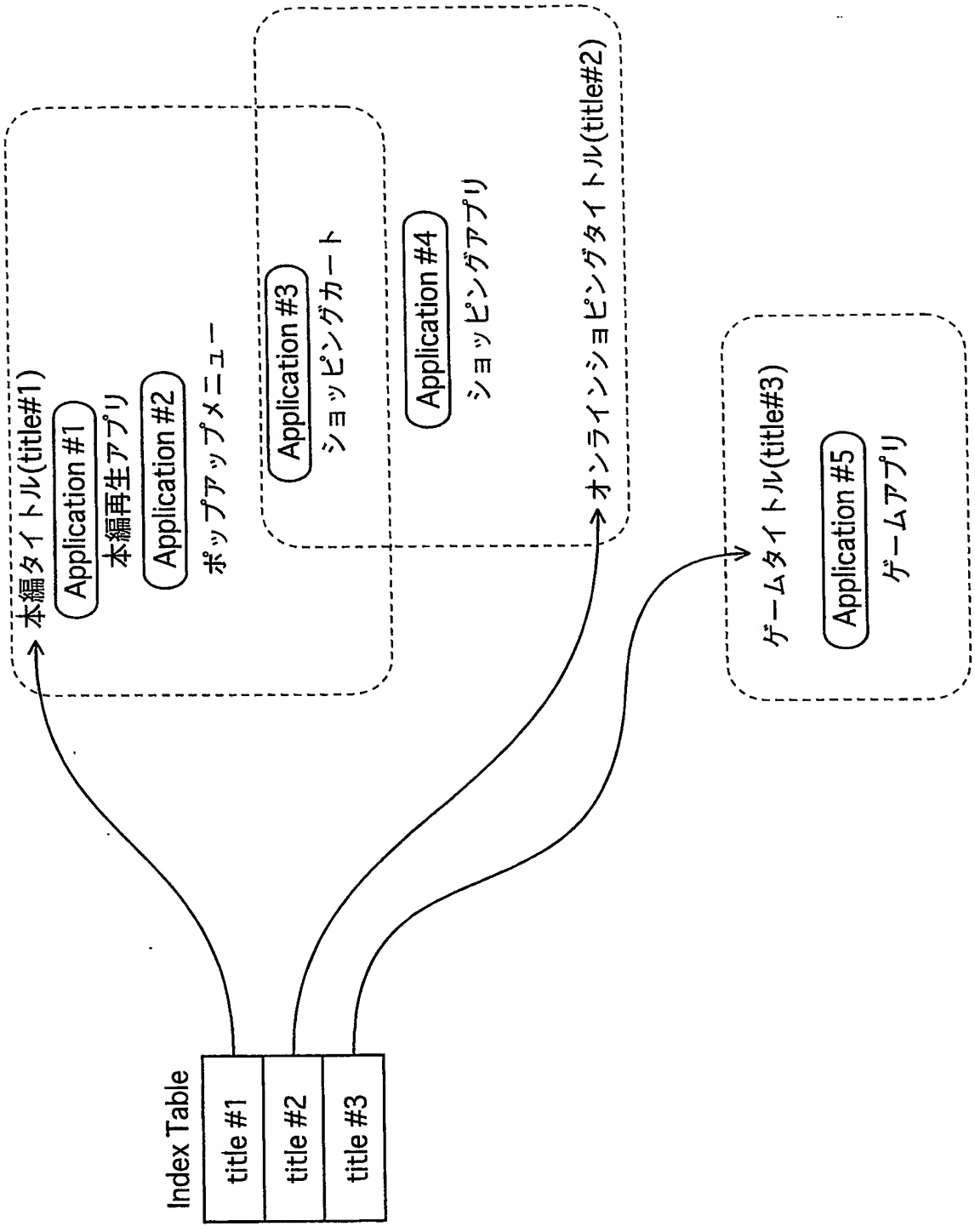

Index Table
title #1
title #2
title #3
本編タイトル(title#1)
Application #1
本編再生アプリ
Application #2
ポップアップメニュー
Application #3
ショッピングカート
Application #4
ショッピングアプリ
オンラインショピングタイトル(title#2)
ゲームタイトル(title#3)
Application #5
ゲームアプリ

図11

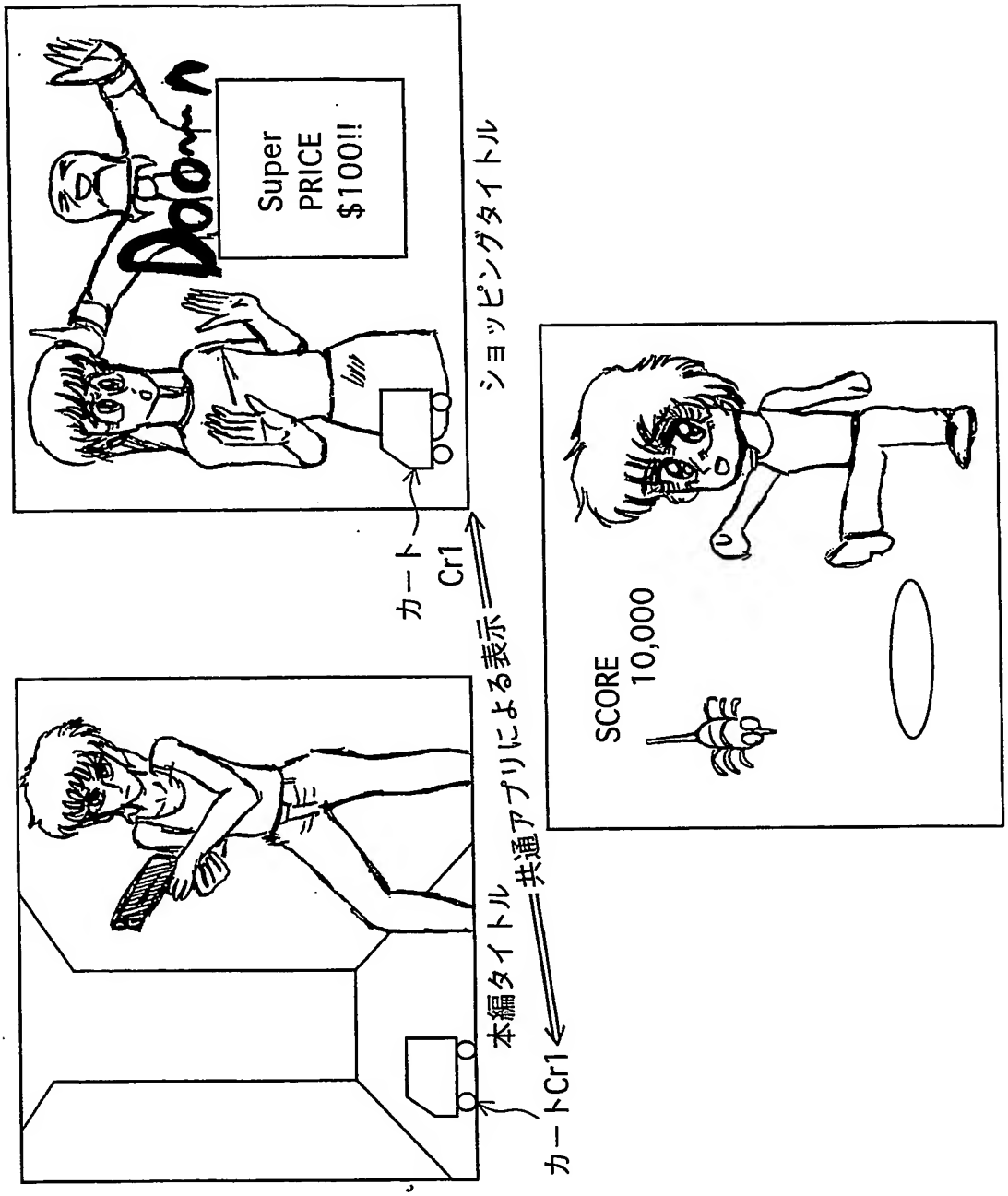
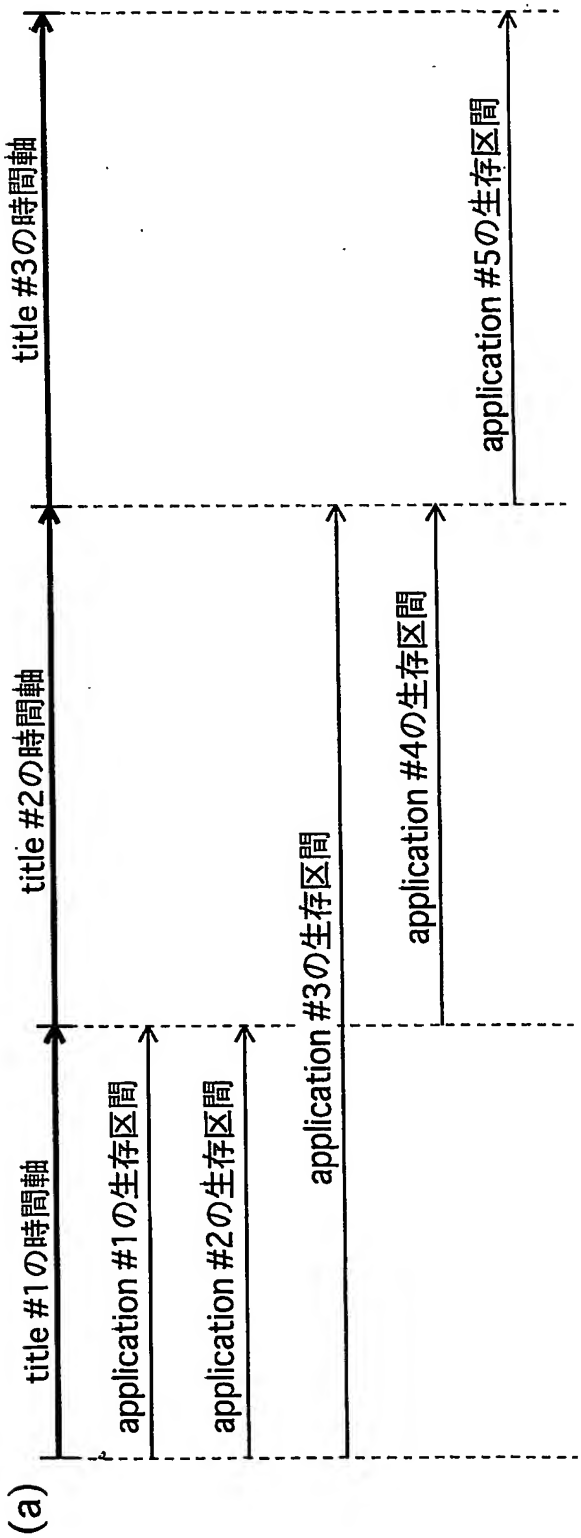

Super
PRICE
$100!!
カート
Cr1
ショッピングタイトル
SCORE
10,000
ゲームタイトル
本編タイトル
共通アプリによる表示
カートCr1

図12



(b)

title #1のアプリケーション管理テーブル

生存区間	アプリケーションID	起動属性
title #1	application #1	
title #1	application #2	
title #1	application #3	

title #2のアプリケーション管理テーブル

生存区間	アプリケーションID	起動属性
title #2	application #3	
title #2	application #4	

title #3のアプリケーション管理テーブル

生存区間	アプリケーションID	起動属性
title #3	application #5	

図13

(a)

title #1のアプリケーション管理テーブル

生存区間	アプリケーションID	起動属性
title #1	application #1	AutoRun
title #1	application #2	Persistent
title #1	application #3	AutoRun

title #2のアプリケーション管理テーブル

生存区間	アプリケーションID	起動属性
title #2	application #3	Persistent

(b)

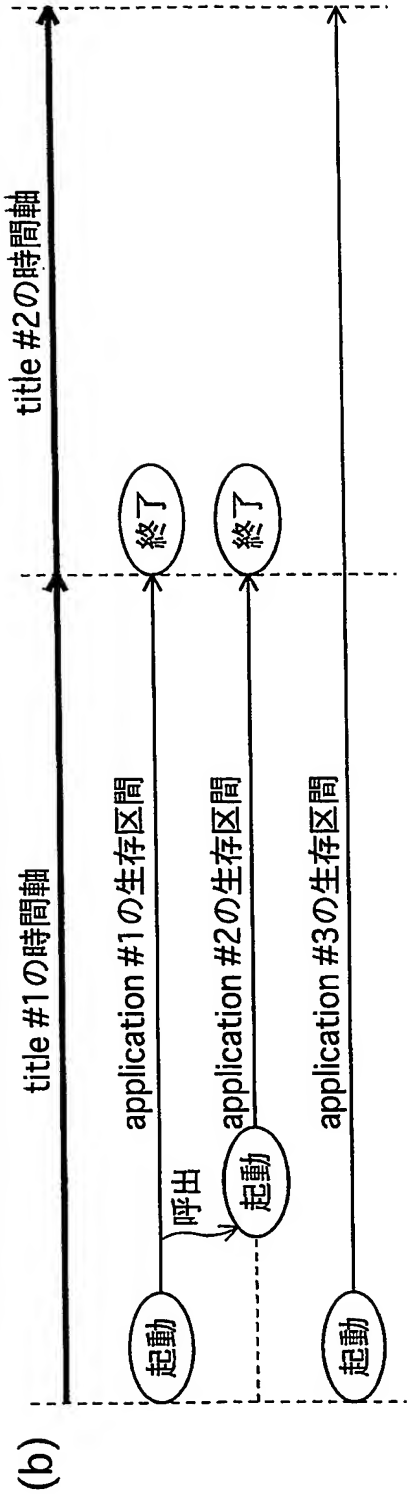


図14

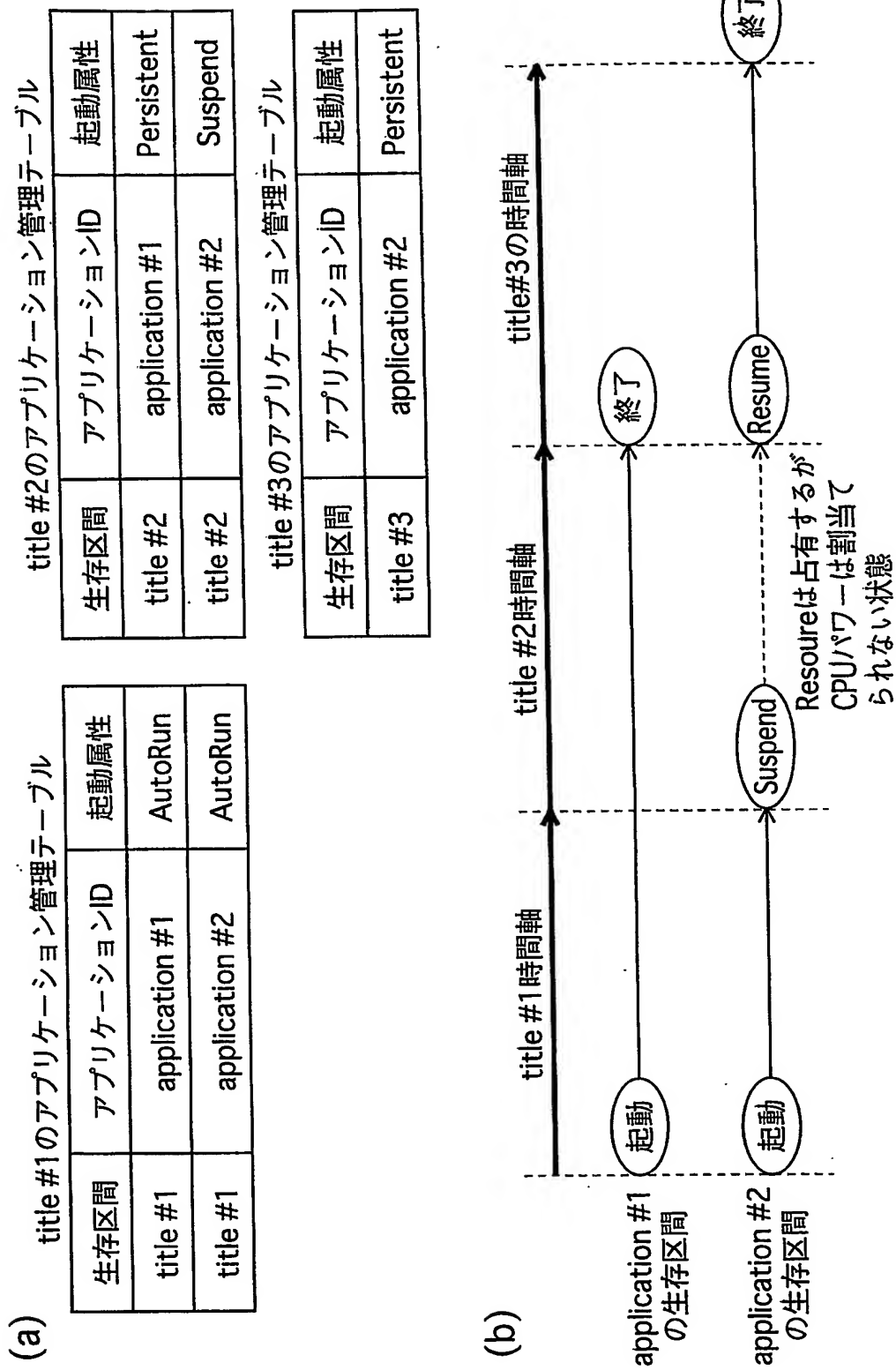

(a)
title #1のアプリケーション管理テーブル
生存区間
アプリケーションID
起動属性
title #1
application #1
AutoRun
title #1
application #2
AutoRun
title #2のアプリケーション管理テーブル
生存区間
アプリケーションID
起動属性
title #2
application #1
Persistent
title #2
application #2
Suspend
title #3のアプリケーション管理テーブル
生存区間
アプリケーションID
起動属性
title #3
application #2
Persistent
(b)
title #1時間軸
title #2時間軸
title#3の時間軸
起動
終了
起動
Suspend
Resume
終了
Resoureは占有するが
CPUパワーは割当て
られない状態
application #1
の生存区間
application #2
の生存区間

図15

起動属性によるアプリケーション状態の変化

直前タイトルにおけるアプリケーションの状態	起動属性		
	Persistent	AutoRun	Suspend
非起動	何もせず、状態継続	アプリケーションを起動する	何もせず、状態継続
起動中	何もせず、状態継続	何もせず、状態継続	サスペンドする
Suspend	レジュームする	レジュームする	何もせず、状態継続

図16

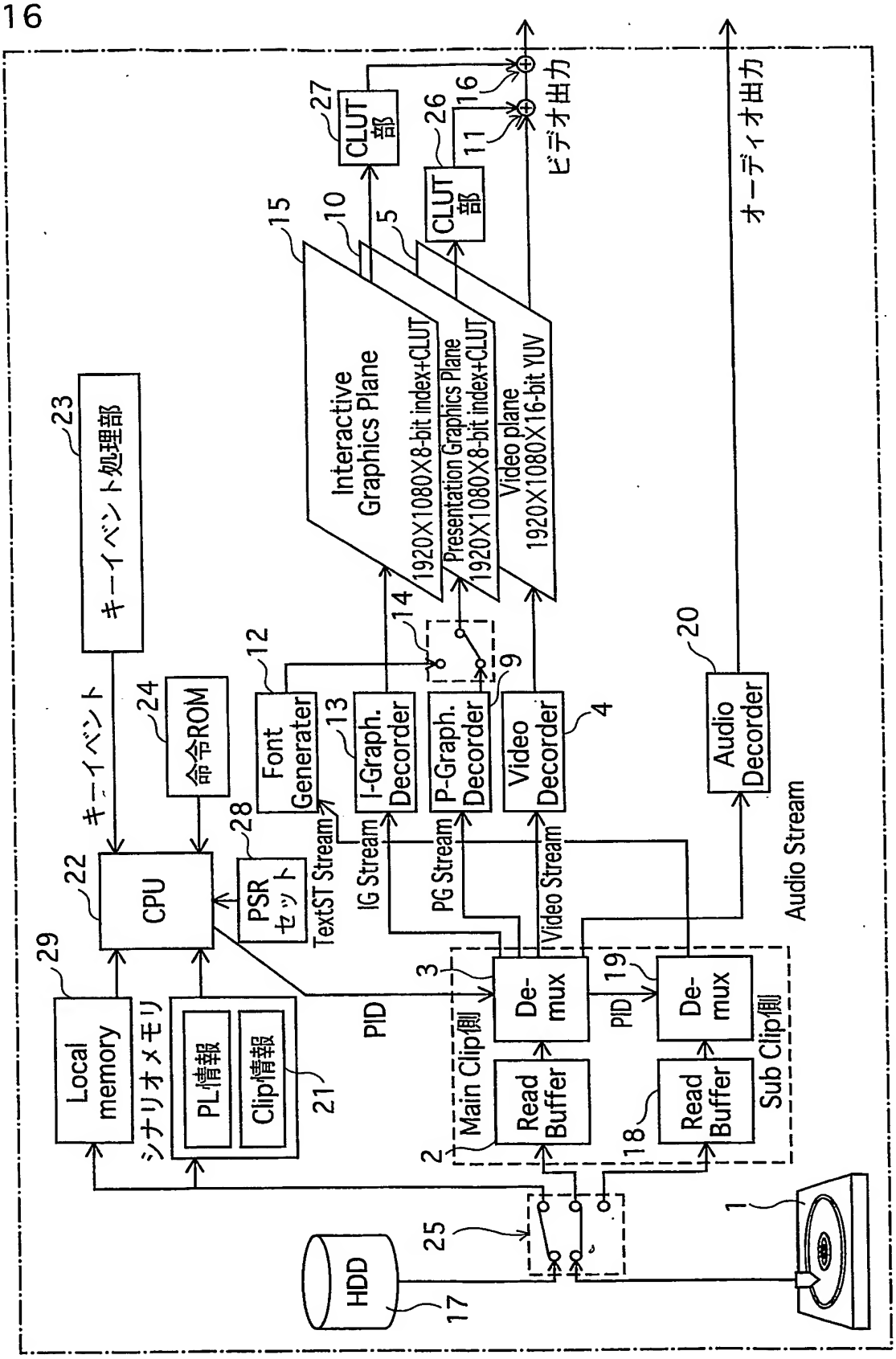

ビデオ出力
オーディオ出力
キーイベント処理部
キーイベント
CPU
命令ROM
PSR セット
Font Generater
I-Graph. Decorder
P-Graph. Decorder
Video Decorder
Audio Decorder
Interactive Graphics Plane
1920X1080X8-bit index+CLUT
Presentation Graphics Plane
1920X1080X8-bit index+CLUT
Video plane
1920X1080X16-bit YUV
CLUT 部
CLUT 部
IG Stream
PG Stream
Video Stream
TextST Stream
Audio Stream
PID
PID
De-mux
De-mux
Read Buffer
Read Buffer
Main Clip側
Sub Clip側
Local memory
シナリオメモリ
PL情報
Clip情報
HDD
1
2
3
4
5
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29

図17

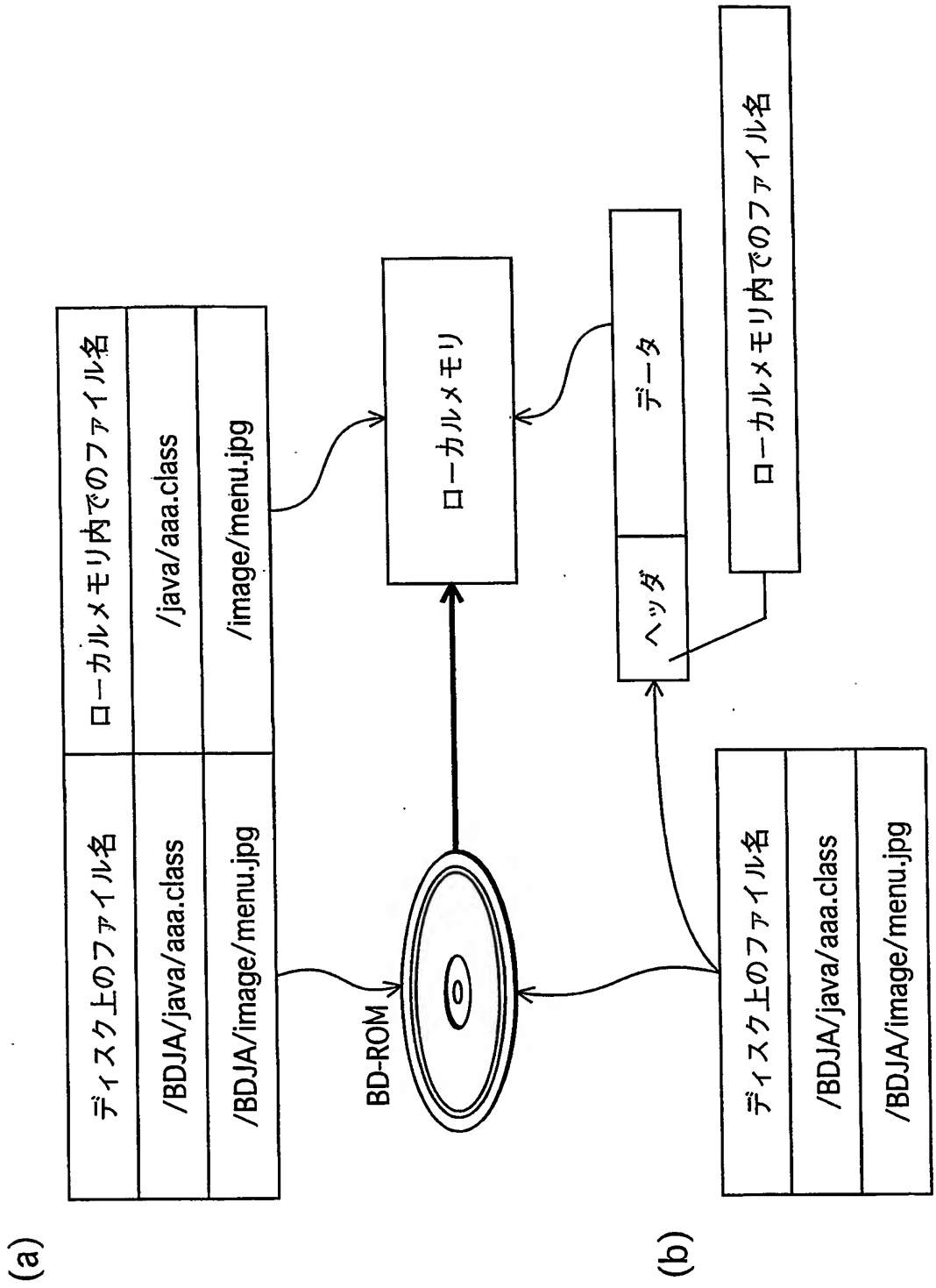

(a)
ディスク上のファイル名
/BDJA/java/aaa.class
/BDJA/image/menu.jpg
BD-ROM
ローカルメモリ内でのファイル名
/java/aaa.class
/image/menu.jpg
ローカルメモリ
(b)
ディスク上のファイル名
/BDJA/java/aaa.class
/BDJA/image/menu.jpg
ヘッダ
データ
ローカルメモリ内でのファイル名

図18

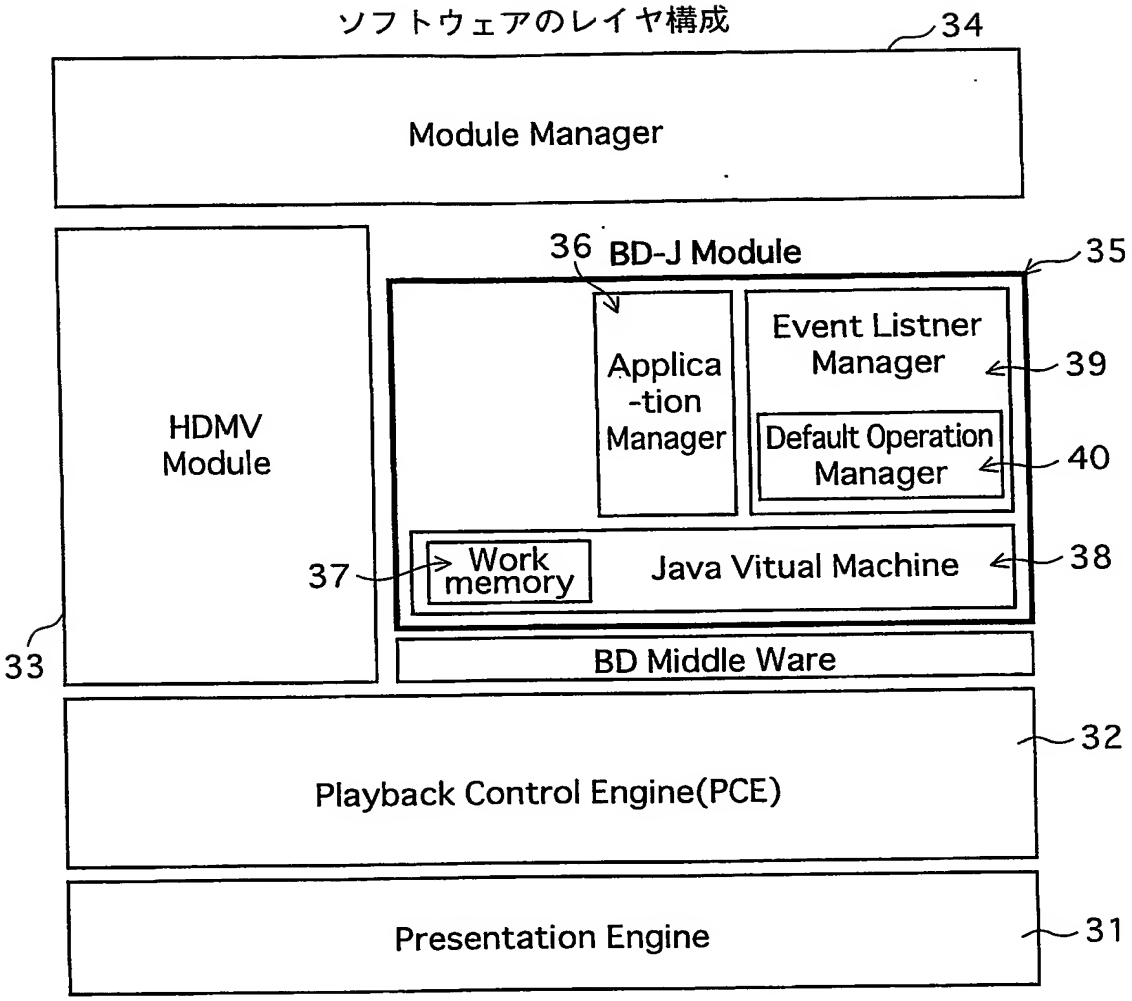

ソフトウェアのレイヤ構成
34
Module Manager
36
BD-J Module
35
Applica
-tion
Manager
Event Listner
Manager
39
HDMV
Module
Default Operation
Manager
40
37
Work
memory
Java Vitual Machine
38
33
BD Middle Ware
32
Playback Control Engine(PCE)
Presentation Engine
31

図19

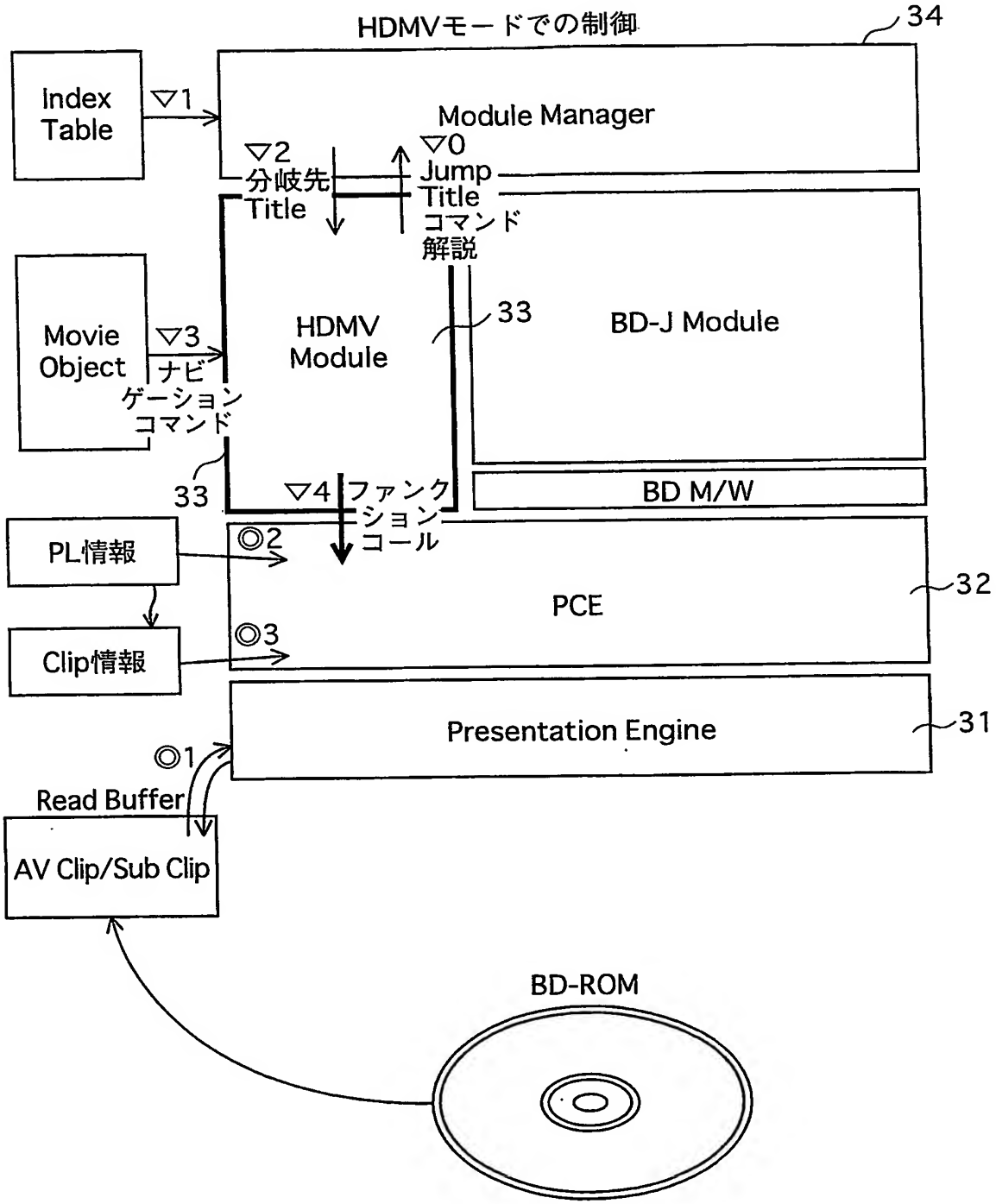
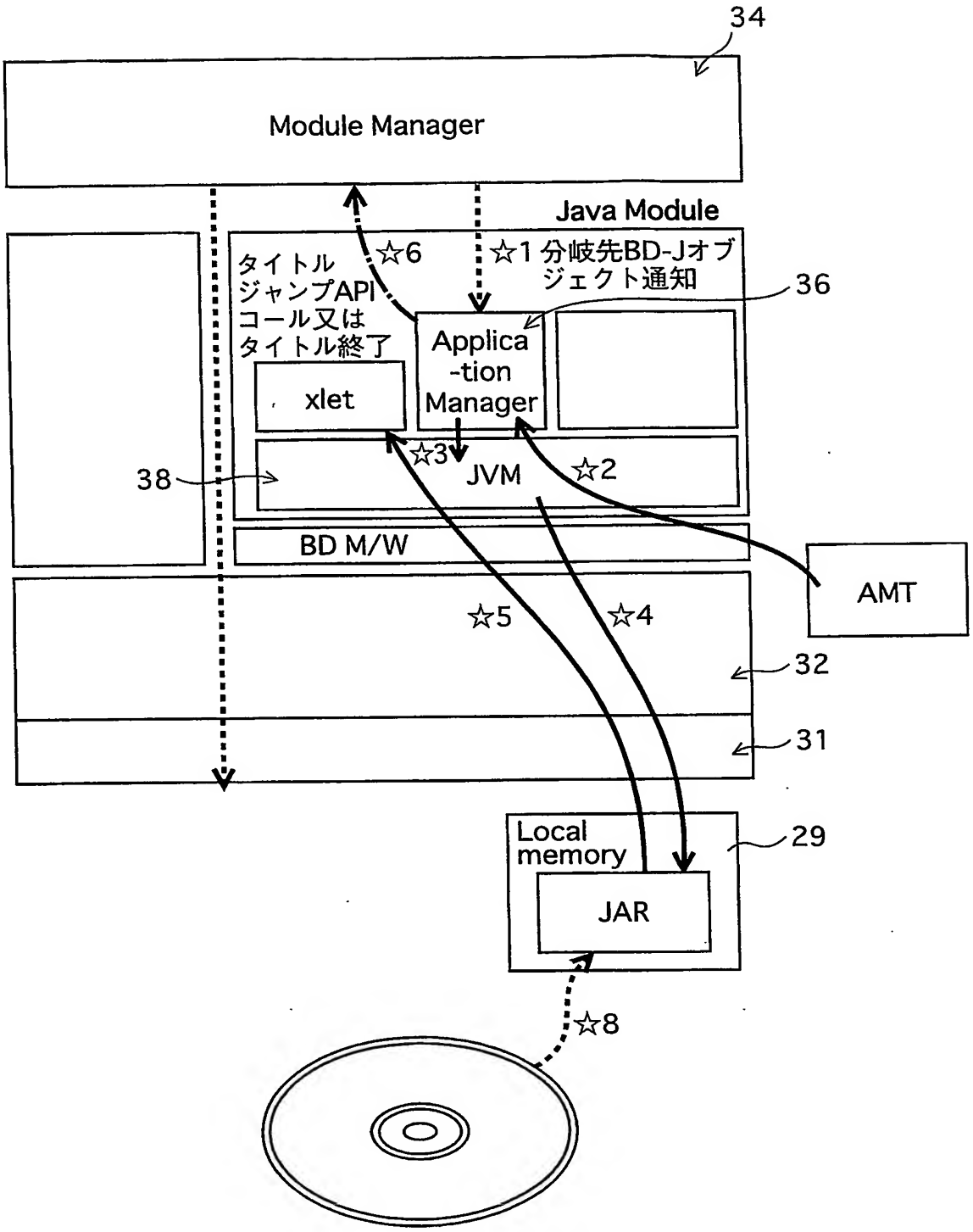

HDMVモードでの制御
34
Index Table
▽1
Module Manager
▽2
分岐先 Title
▽0
Jump Title
コマンド解説
Movie Object
▽3
ナビゲーションコマンド
HDMV Module
33
BD-J Module
33
▽4 ファンクションコール
BD M/W
PL情報
◎2
PCE
32
Clip情報
◎3
Presentation Engine
31
◎1
Read Buffer
AV Clip/Sub Clip
BD-ROM

図20



xletのロード及び起動制御

図21

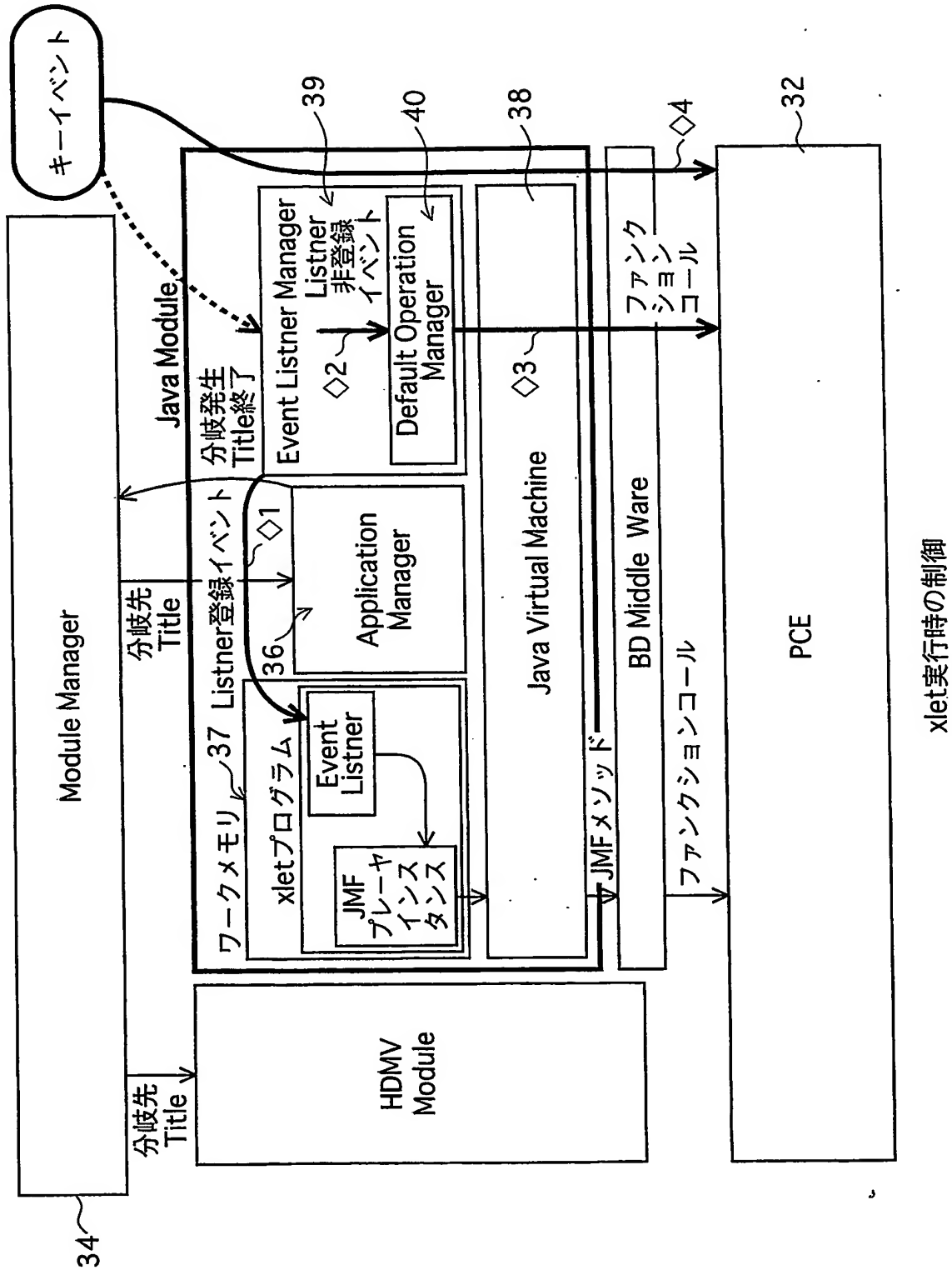

キーイベント
Module Manager
34
分岐先 Title
分岐先 Title
Java Module
ワークメモリ
37
xletプログラム
Event Listner
JMF プレーヤ インスタンス
Listner登録イベント
◇1
36
Application Manager
分岐発生 Title終了
Event Listner Manager
39
Listner 非登録 イベント
◇2
40
Default Operation Manager
◇3
38
Java Virtual Machine
JMFメソッド
◇4
BD Middle Ware
ファンクションコール
ファンクションコール
32
PCE
HDMV Module
xlet実行時の制御

図22

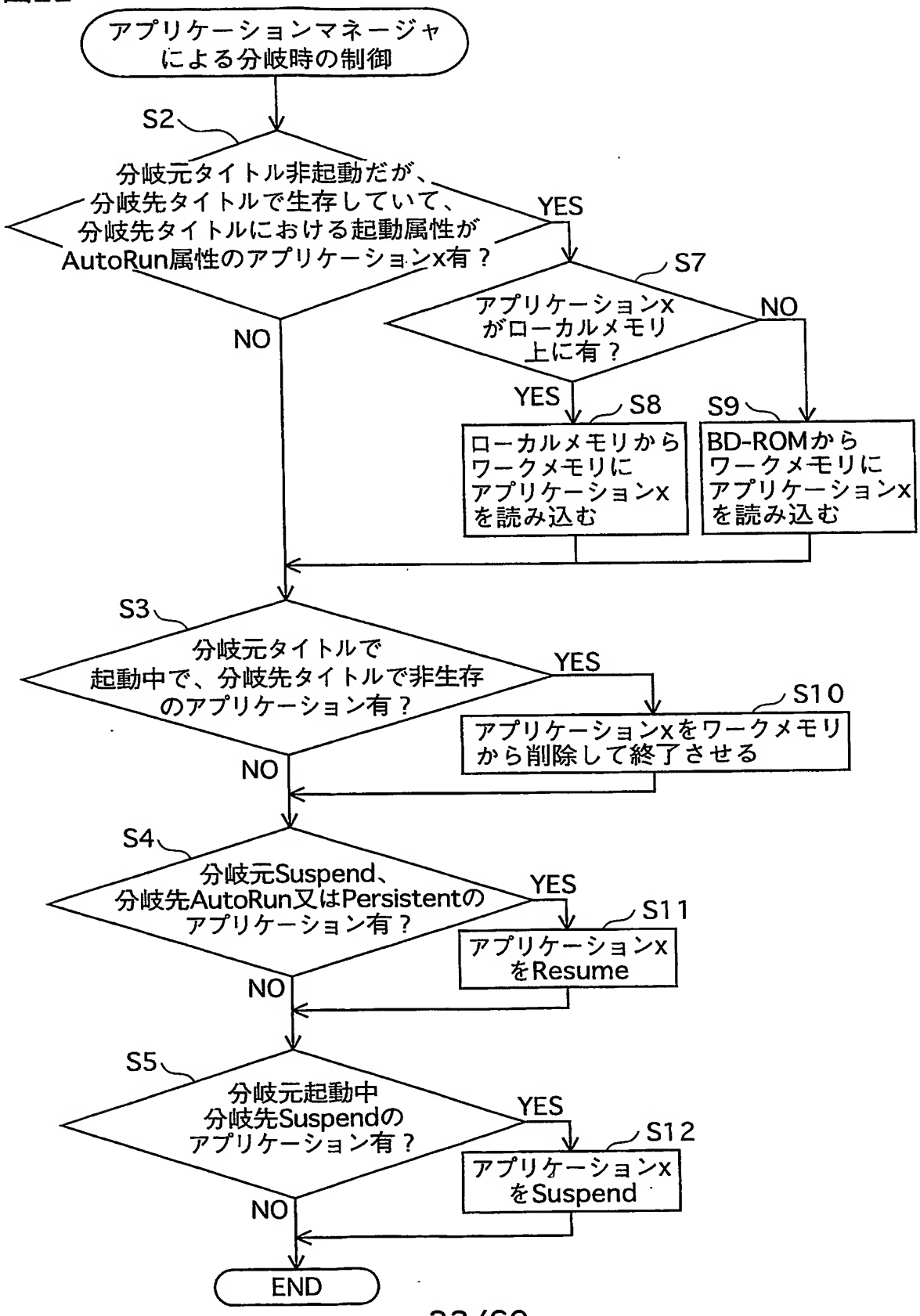

アプリケーションマネージャ
による分岐時の制御
S2
分岐元タイトル非起動だが、
分岐先タイトルで生存していて、
分岐先タイトルにおける起動属性が
AutoRun属性のアプリケーションx有？
YES
NO
S7
アプリケーションx
がローカルメモリ
上に有？
NO
YES
S8
ローカルメモリから
ワークメモリに
アプリケーションx
を読み込む
S9
BD-ROMから
ワークメモリに
アプリケーションx
を読み込む
S3
分岐元タイトルで
起動中で、分岐先タイトルで非生存
のアプリケーション有？
YES
NO
S10
アプリケーションxをワークメモリ
から削除して終了させる
S4
分岐元Suspend、
分岐先AutoRun又はPersistentの
アプリケーション有？
YES
NO
S11
アプリケーションx
をResume
S5
分岐元起動中
分岐先Suspendの
アプリケーション有？
YES
NO
S12
アプリケーションx
をSuspend
END

図23

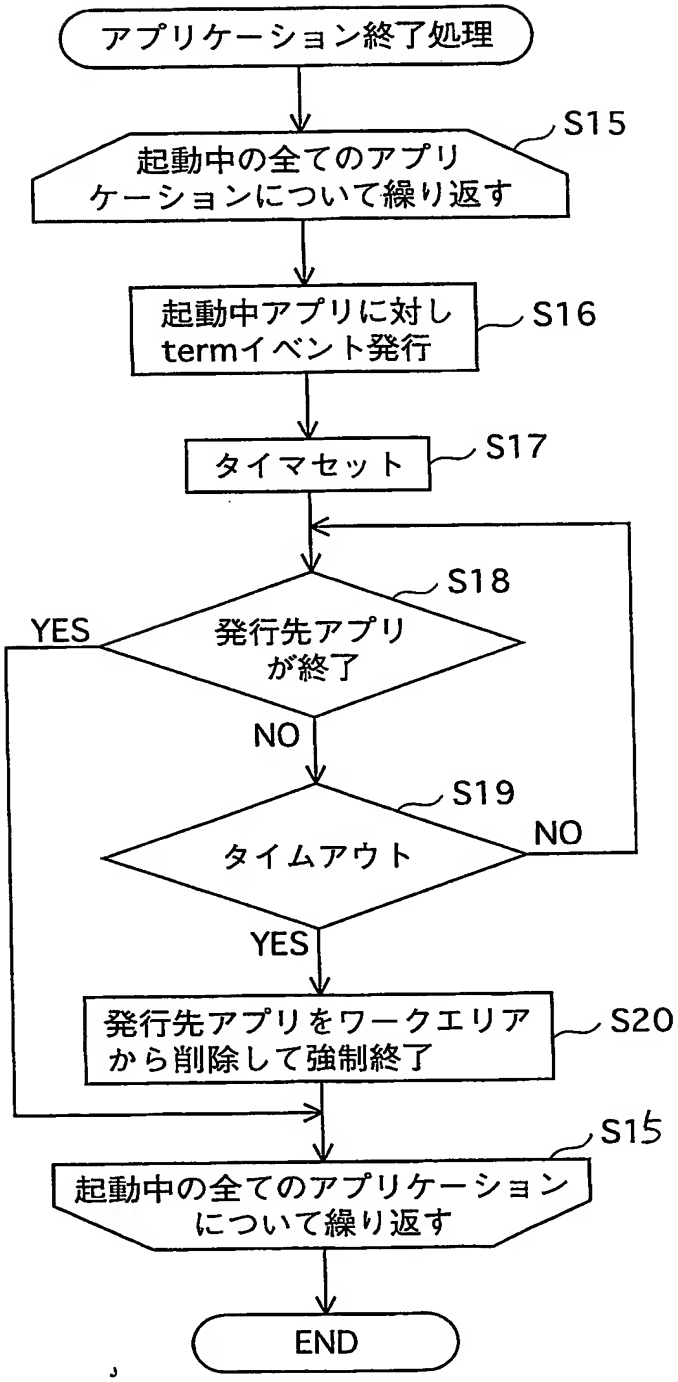

アプリケーション終了処理
S15
起動中の全てのアプリケーションについて繰り返す
起動中アプリに対しtermイベント発行
S16
タイマセット
S17
S18
YES
発行先アプリが終了
NO
S19
タイムアウト
NO
YES
発行先アプリをワークエリアから削除して強制終了
S20
S15
起動中の全てのアプリケーションについて繰り返す
END

図24

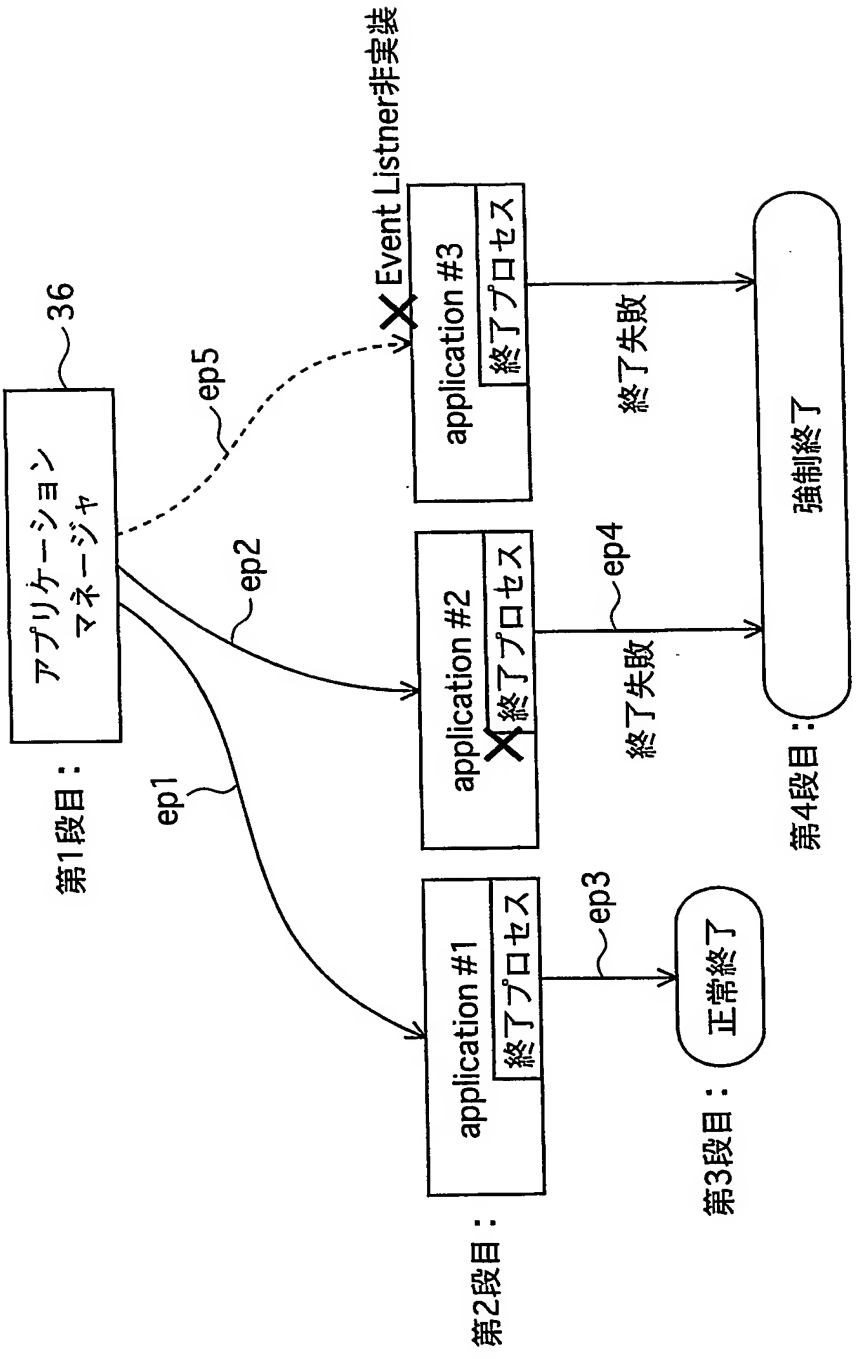

36
アプリケーション
マネージャ
第1段目：
ep1
ep2
ep5
Event Listner非実装
application #1
終了プロセス
第2段目：
ep3
正常終了
第3段目：
application #2
終了プロセス
終了失敗
ep4
application #3
終了プロセス
終了失敗
強制終了
第4段目：

図25

(a)

アプリケーション管理テーブル

生存区間	アプリケーションID	起動属性	起動優先度
title #1	application #1	AutoRun	255
title #1：chapter #2-#3	application #2	AutoRun	128
title #1：chapter #4-#6	application #3	AutoRun	128

(b)

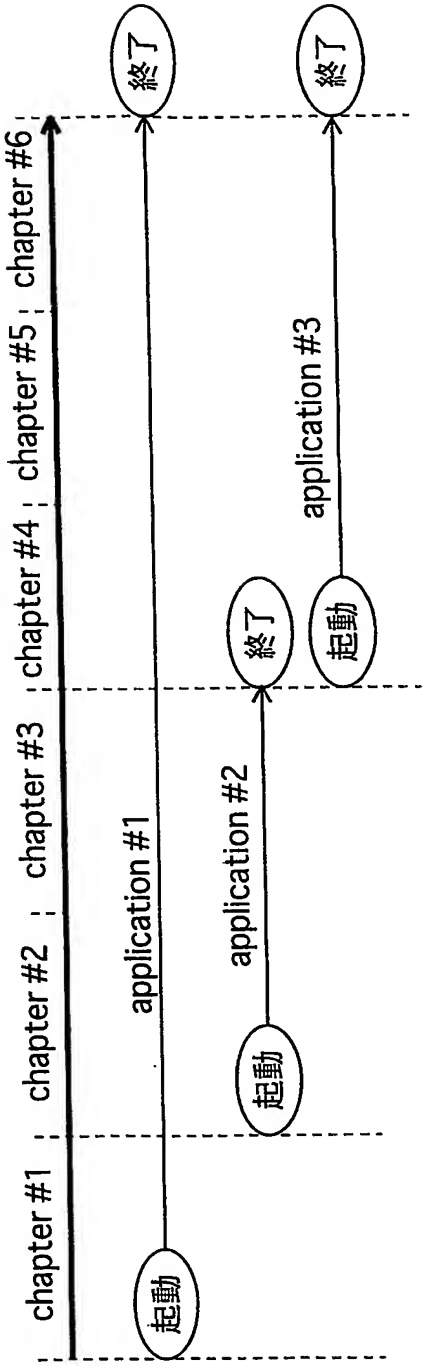


図26

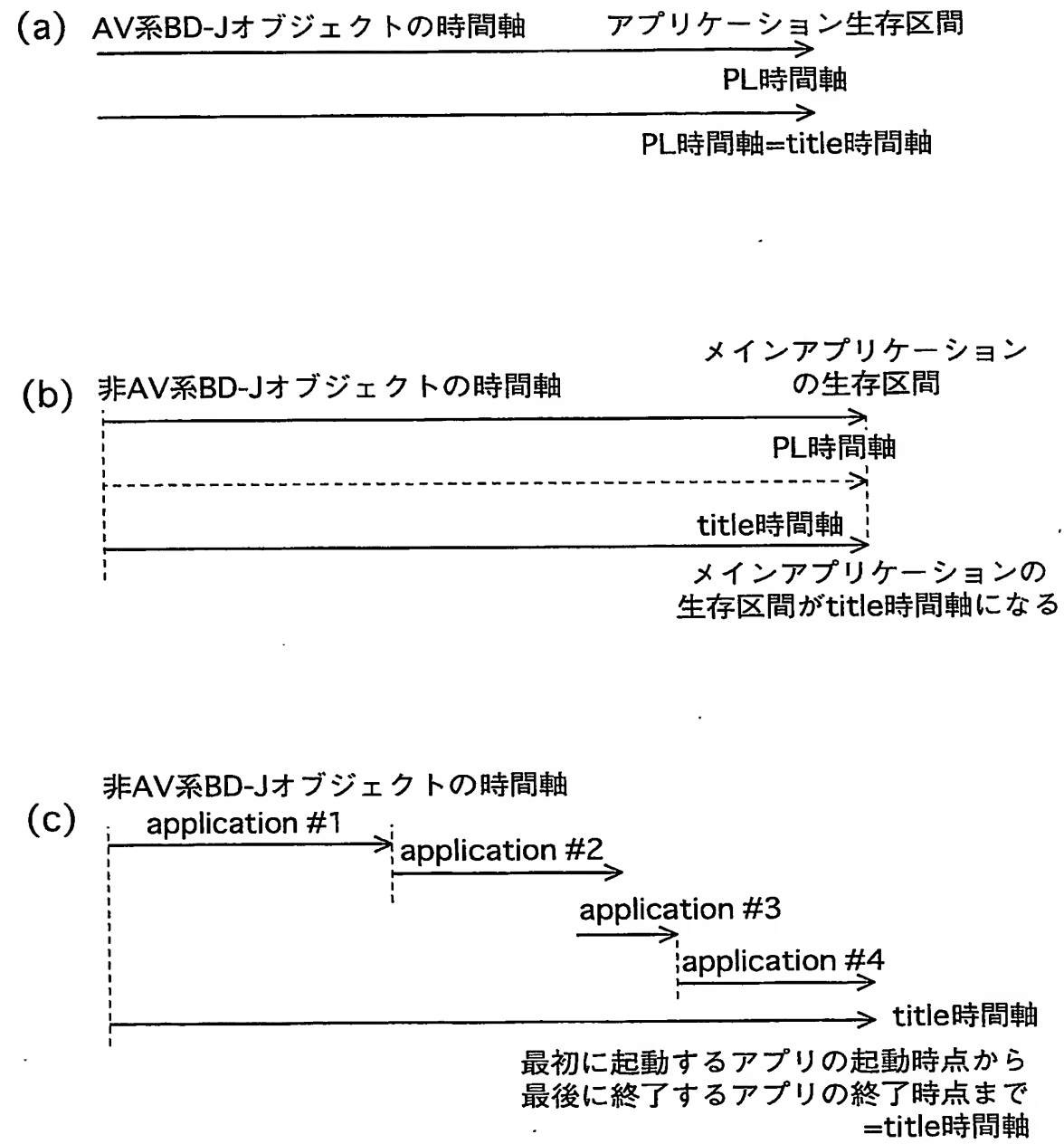

(a)
AV系BD-Jオブジェクトの時間軸
アプリケーション生存区間
PL時間軸
PL時間軸=title時間軸
(b)
非AV系BD-Jオブジェクトの時間軸
メインアプリケーション
の生存区間
PL時間軸
title時間軸
メインアプリケーションの
生存区間がtitle時間軸になる
(c)
非AV系BD-Jオブジェクトの時間軸
application #1
application #2
application #3
application #4
title時間軸
最初に起動するアプリの起動時点から
最後に終了するアプリの終了時点まで
=title時間軸

図27

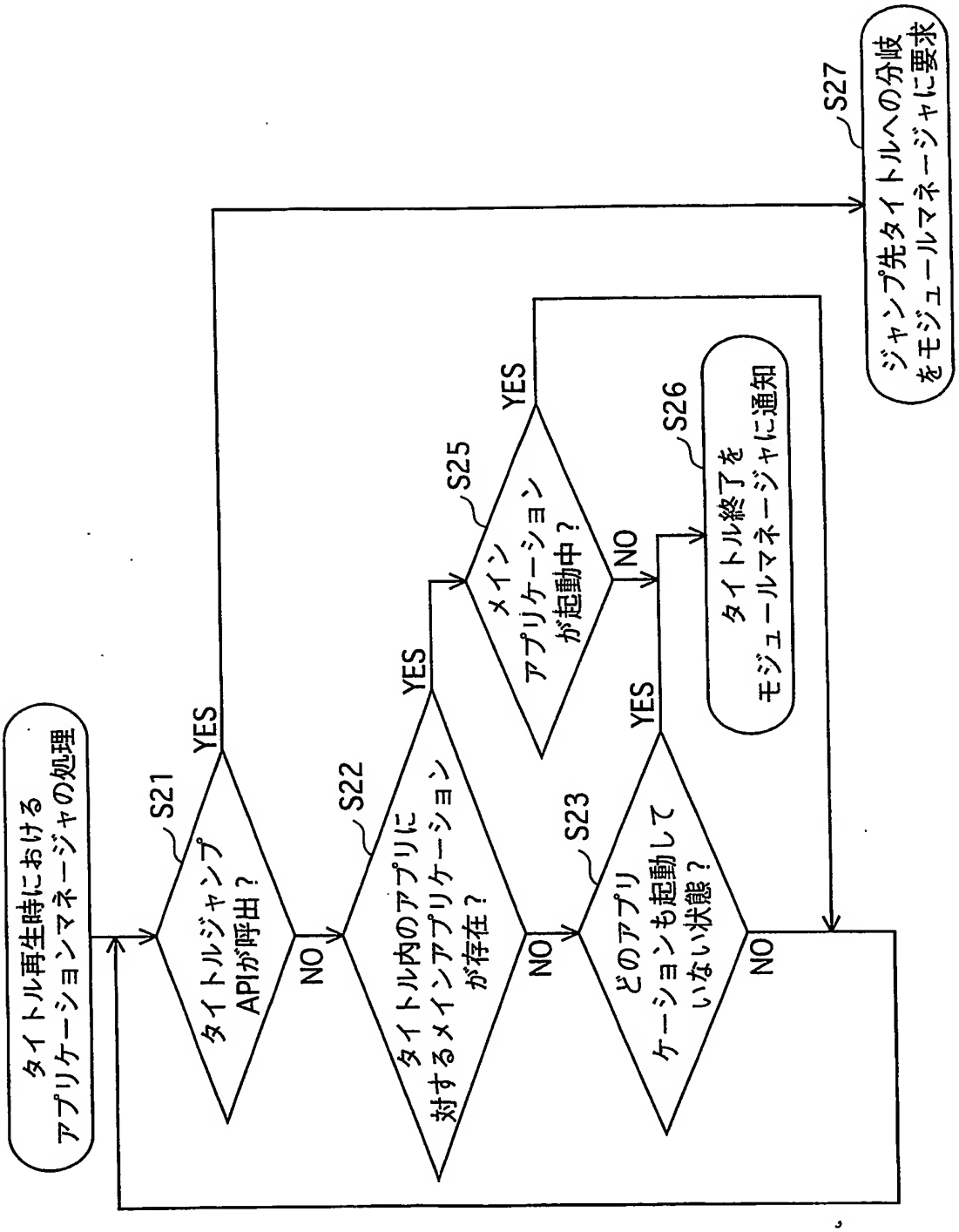

タイトル再生時における
アプリケーションマネージャの処理
S21
タイトルジャンプ
APIが呼出？
YES
NO
S22
タイトル内のアプリに
対するメインアプリケーション
が存在？
YES
NO
S23
どのアプリ
ケーションも起動して
いない状態？
YES
NO
S25
メイン
アプリケーション
が起動中？
YES
NO
S26
タイトル終了を
モジュールマネージャに通知
S27
ジャンプ先タイトルへの分岐
をモジュールマネージャに要求

図28

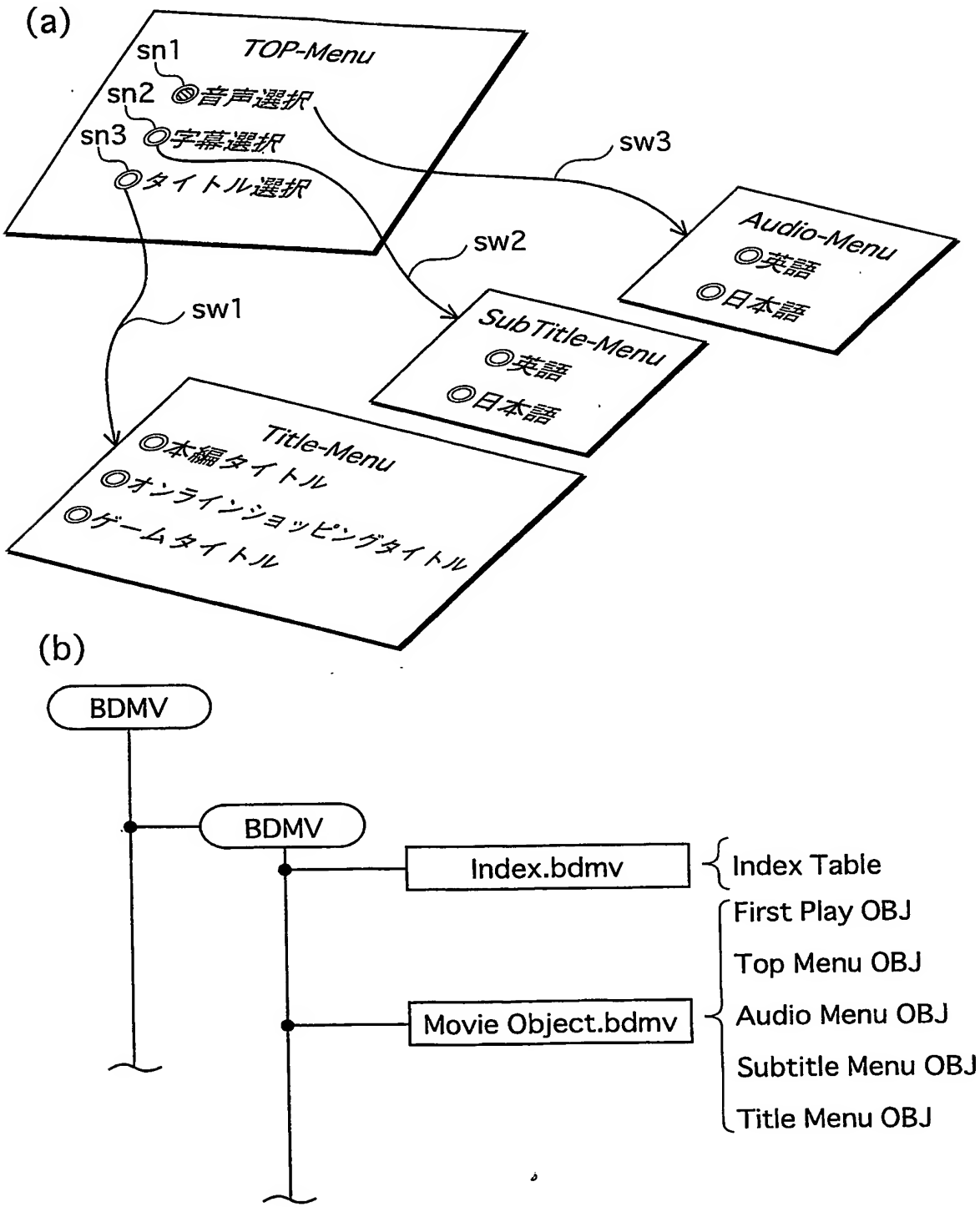

(a)
sn1
TOP-Menu
sn2 ◎音声選択
sn3 ◎字幕選択
◎タイトル選択
sw3
sw2
sw1
Audio-Menu
◎英語
◎日本語
SubTitle-Menu
◎英語
◎日本語
Title-Menu
◎本編タイトル
◎オンラインショッピングタイトル
◎ゲームタイトル
(b)
BDMV
BDMV
Index.bdmv
Index Table
Movie Object.bdmv
First Play OBJ
Top Menu OBJ
Audio Menu OBJ
Subtitle Menu OBJ
Title Menu OBJ

図29

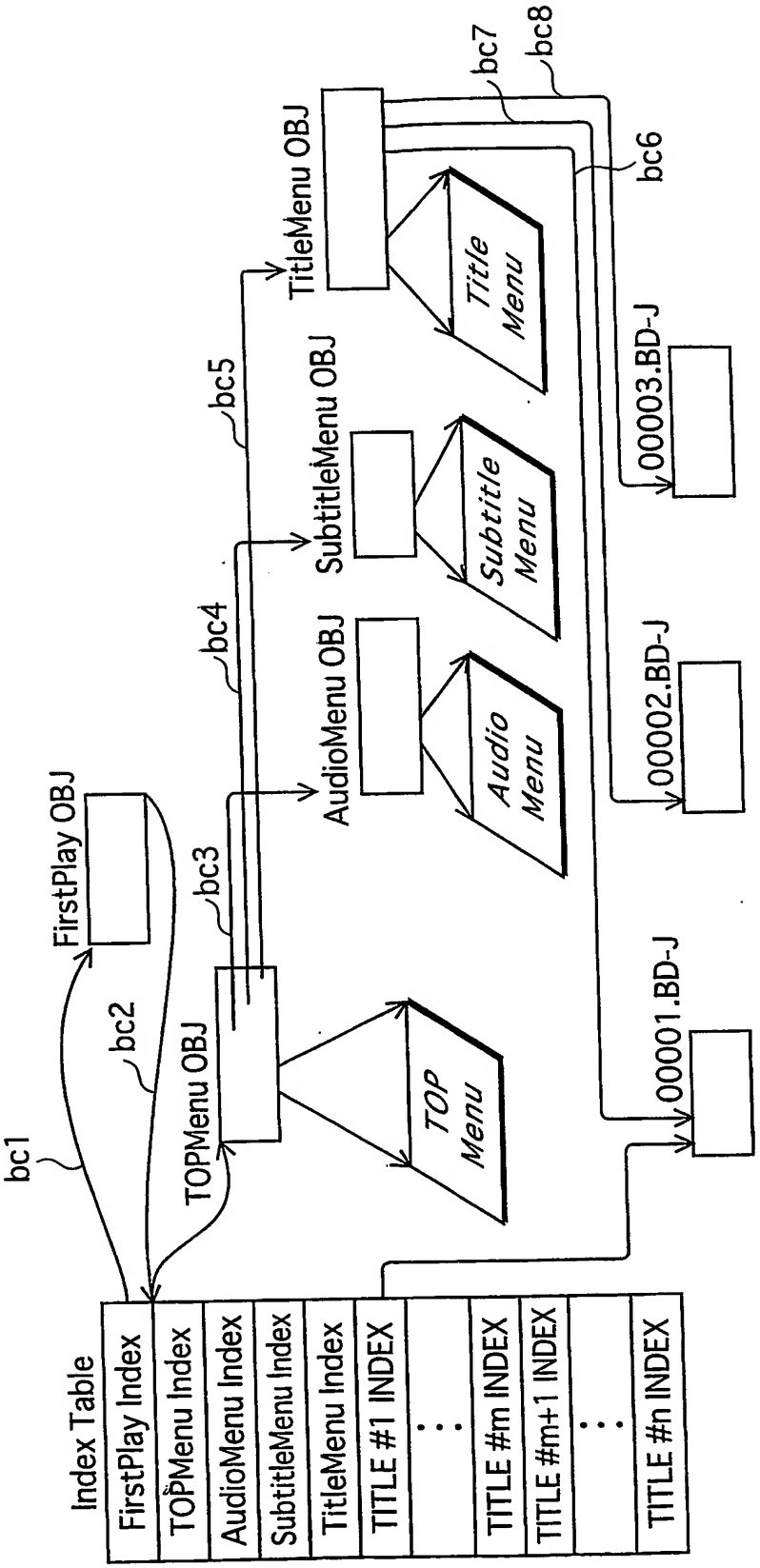
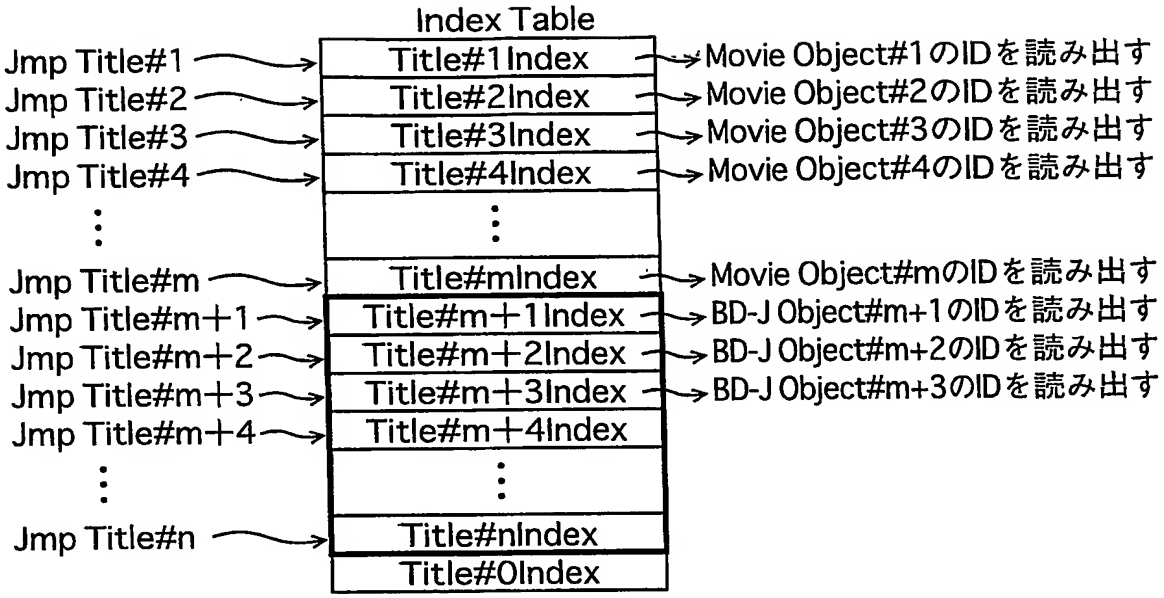

Index Table
FirstPlay Index
TOPMenu Index
AudioMenu Index
SubtitleMenu Index
TitleMenu Index
TITLE #1 INDEX
TITLE #m INDEX
TITLE #m+1 INDEX
TITLE #n INDEX
bc1
bc2
bc3
bc4
bc5
bc6
bc7
bc8
FirstPlay OBJ
TOPMenu OBJ
AudioMenu OBJ
SubtitleMenu OBJ
TitleMenu OBJ
TOP Menu
Audio Menu
Subtitle Menu
Title Menu
00001.BD-J
00002.BD-J
00003.BD-J

図30

(a)



(b) BD-Jオブジェクト実行時の例外処理

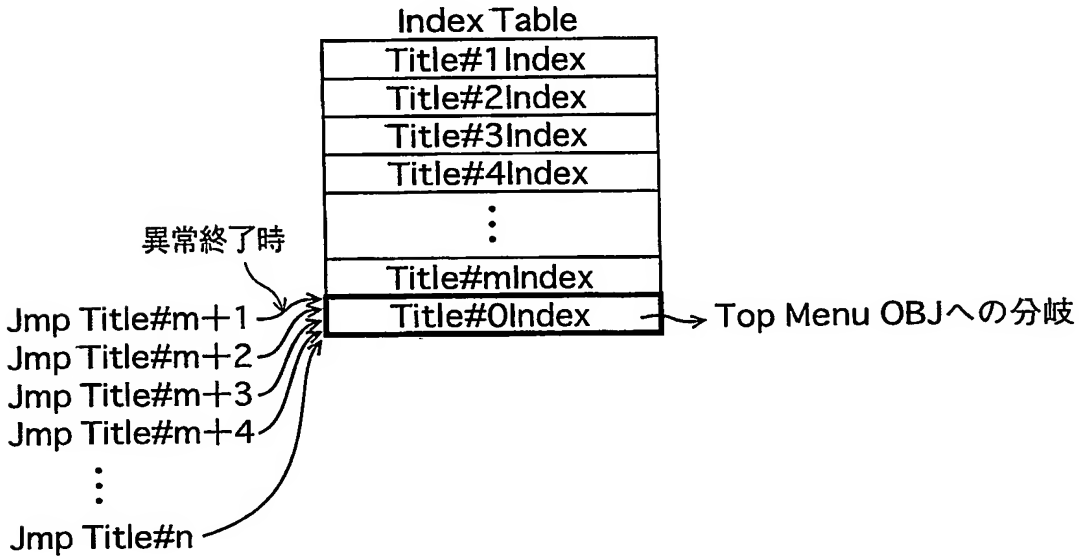


図31

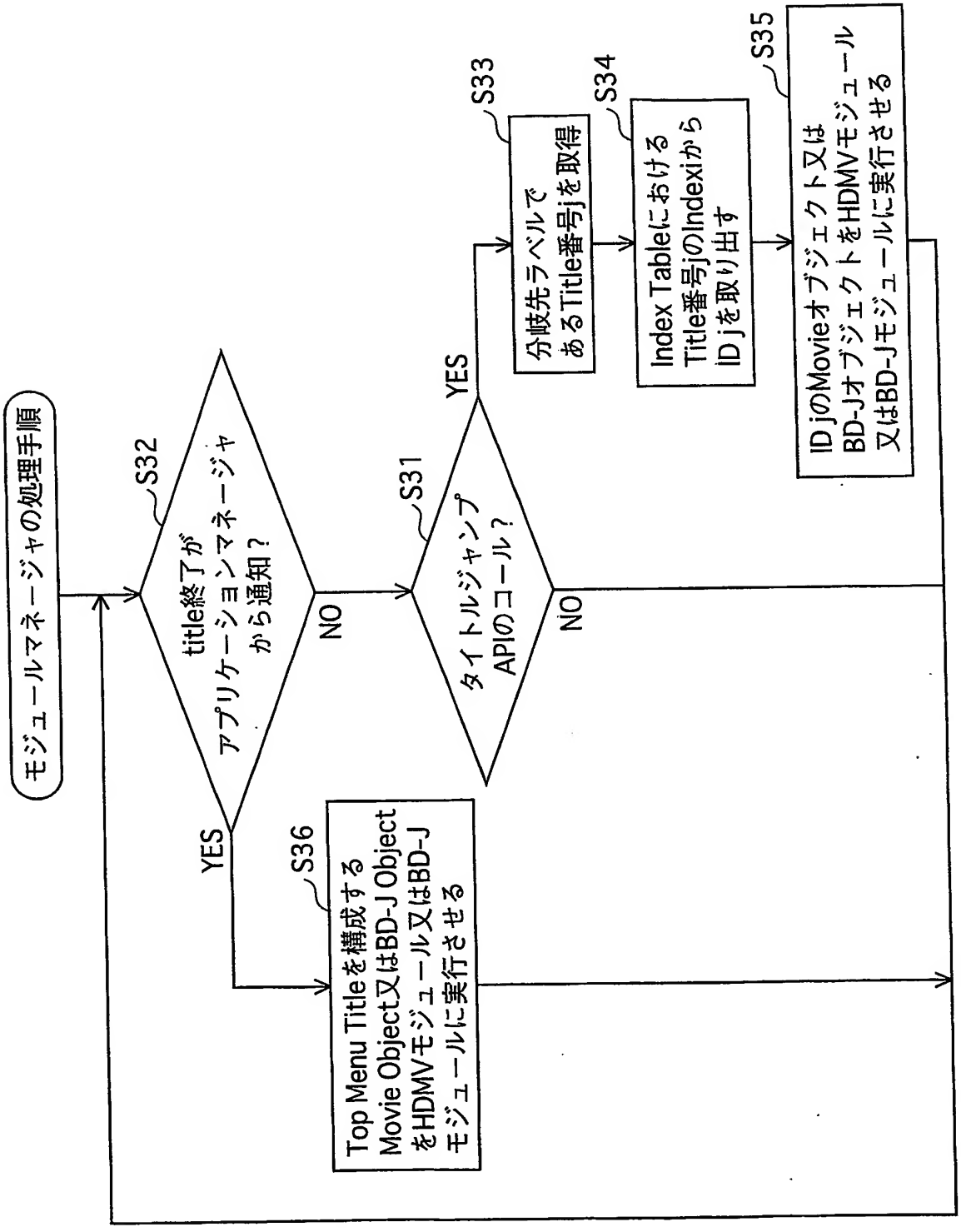

モジュールマネージャの処理手順
S32
title終了が
アプリケーションマネージャ
から通知？
YES
NO
S31
タイトルジャンプ
APIのコール？
YES
NO
S33
分岐先ラベルで
あるTitle番号jを取得
S34
Index Tableにおける
Title番号jのIndexiから
IDjを取り出す
S35
IDjのMovieオブジェクト又は
BD-Jオブジェクトを HDMVモジュール
又はBD-Jモジュールに実行させる
S36
Top Menu Titleを構成する
Movie Object又はBD-J Object
をHDMVモジュール又はBD-J
モジュールに実行させる

図32

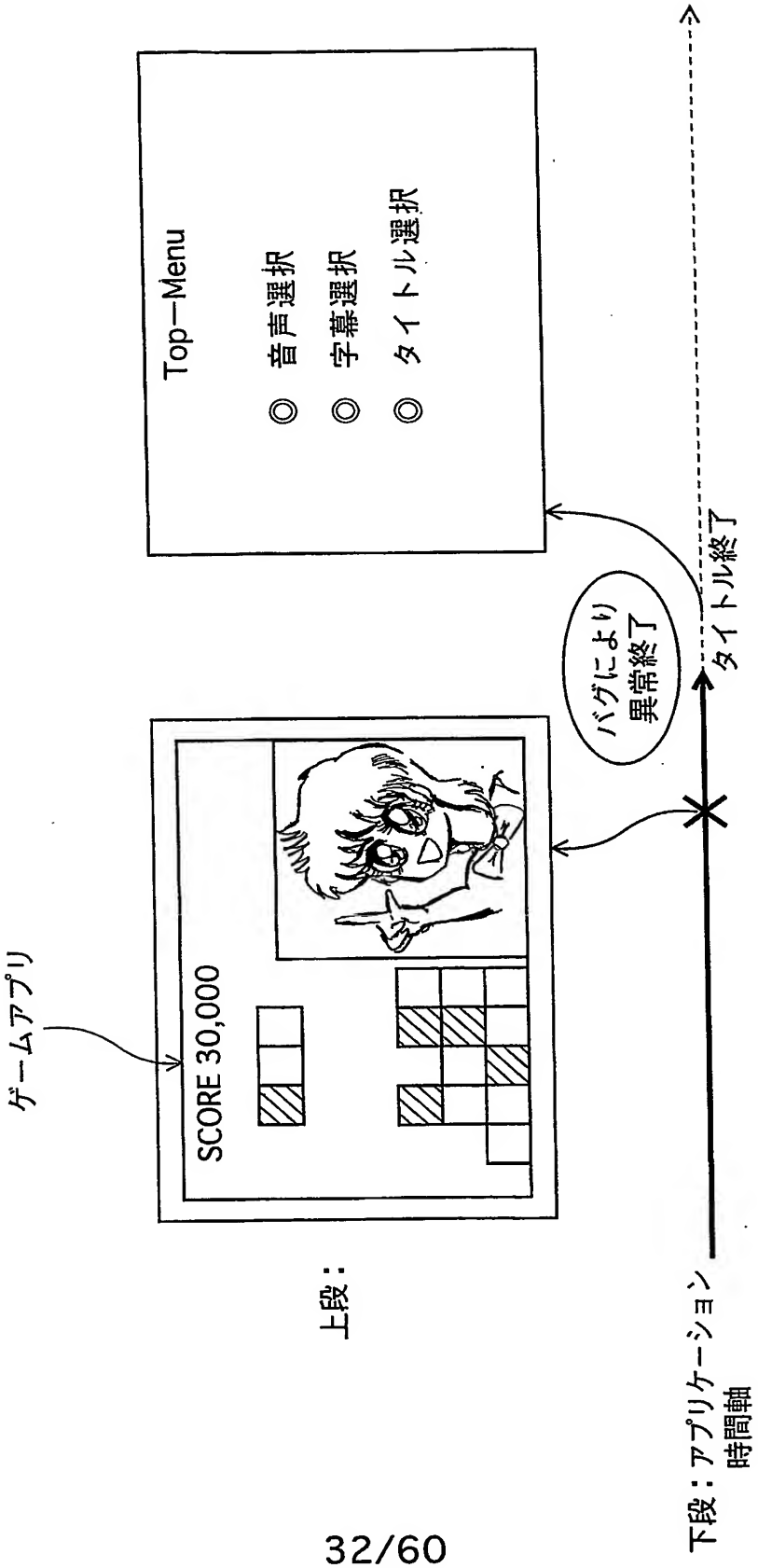

Top—Menu
◎ 音声選択
◎ 字幕選択
◎ タイトル選択
SCORE 30,000
ゲームアプリ
バグにより
異常終了
タイトル終了
上段：
下段：アプリケーション
時間軸

図33

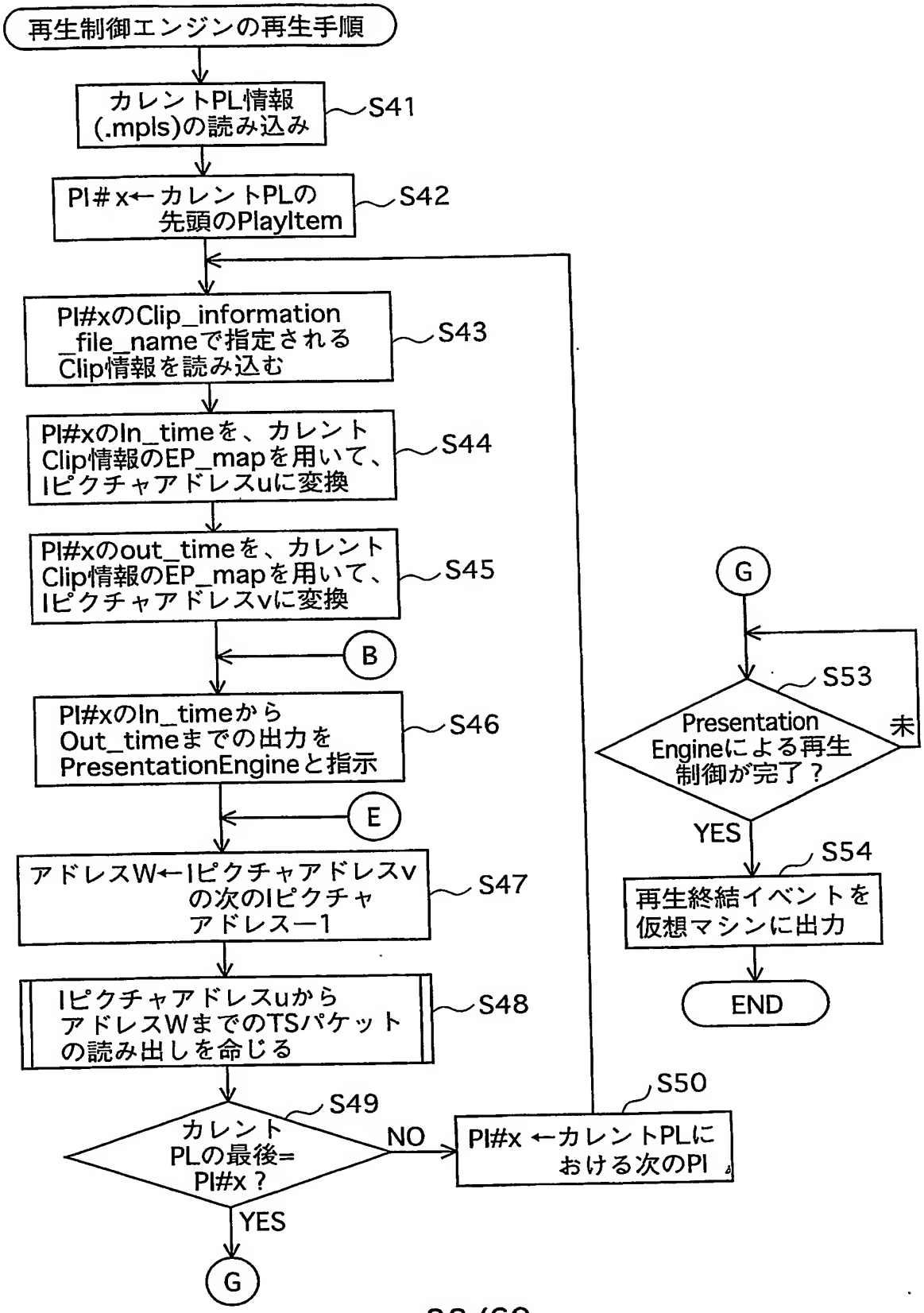

再生制御エンジンの再生手順
カレントPL情報(.mpls)の読み込み
S41
PI#xカレントPLの先頭のPlayItem
S42
PI#xのClip_information_file_nameで指定されるClip情報を読み込む
S43
PI#xのIn_timeを、カレントClip情報のEP_mapを用いて、Iピクチャアドレスuに変換
S44
PI#xのout_timeを、カレントClip情報のEP_mapを用いて、Iピクチャアドレスvに変換
S45
B
PI#xのIn_timeからOut_timeまでの出力をPresentationEngineと指示
S46
E
アドレスW←Iピクチャアドレスvの次のIピクチャアドレス−1
S47
IピクチャアドレスuからアドレスWまでのTSパケットの読み出しを命じる
S48
カレントPLの最後=PI#x？
S49
NO
YES
PI#x←カレントPLにおける次のPI
S50
G
G
Presentation Engineによる再生制御が完了？
S53
未
YES
再生終結イベントを仮想マシンに出力
S54
END

図34

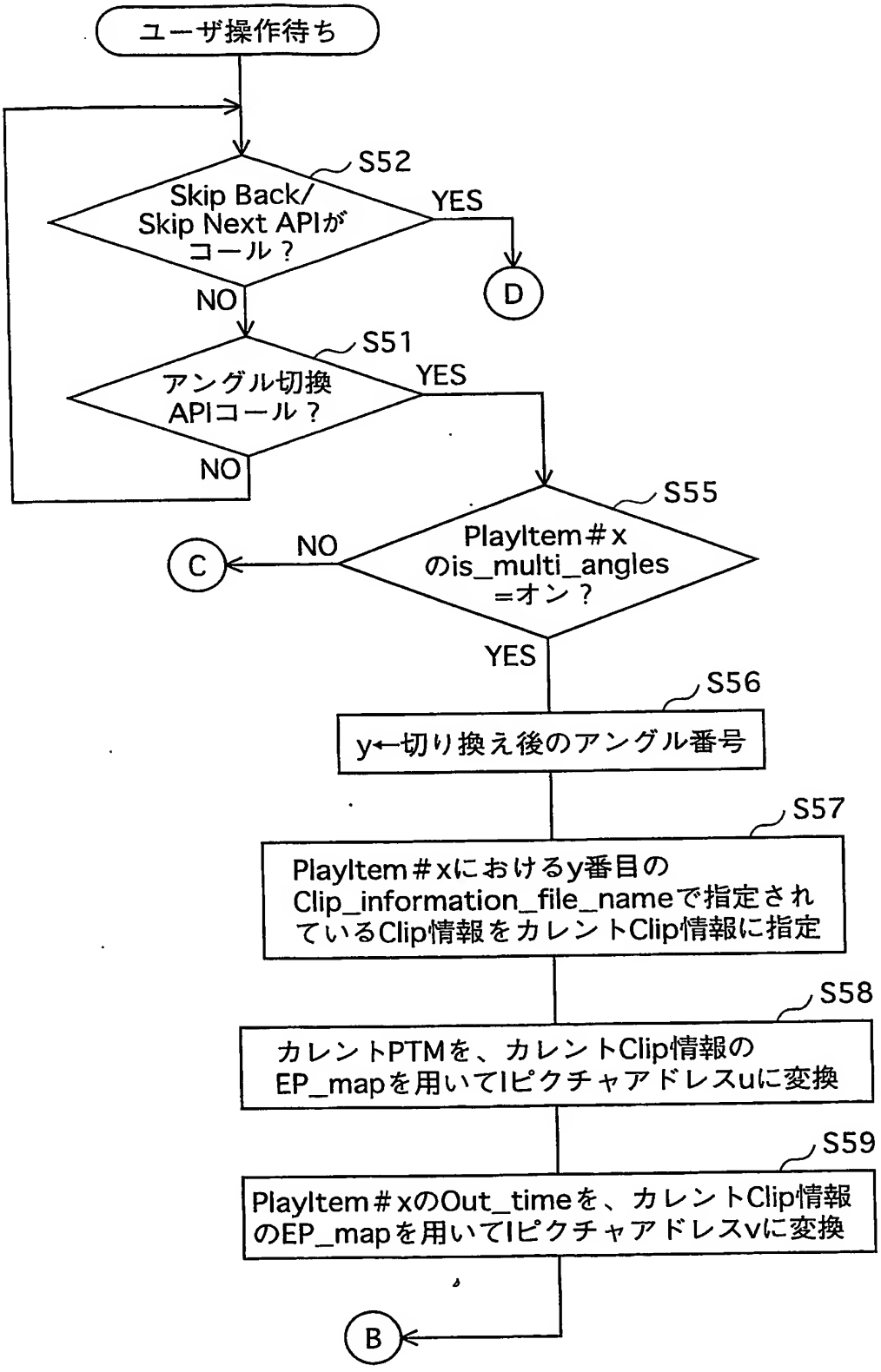

ユーザ操作待ち
S52
Skip Back/
Skip Next APIが
コール？
YES
D
NO
S51
アングル切換
APIコール？
YES
NO
S55
PlayItem#x
のis_multi_angles
=オン？
NO
C
YES
S56
y←切り換え後のアングル番号
S57
PlayItem#xにおけるy番目の
Clip_information_file_nameで指定され
ているClip情報をカレントClip情報に指定
S58
カレントPTMを、カレントClip情報の
EP_mapを用いてIピクチャアドレスuに変換
S59
PlayItem#xのOut_timeを、カレントClip情報
のEP_mapを用いてIピクチャアドレスvに変換
B

図35

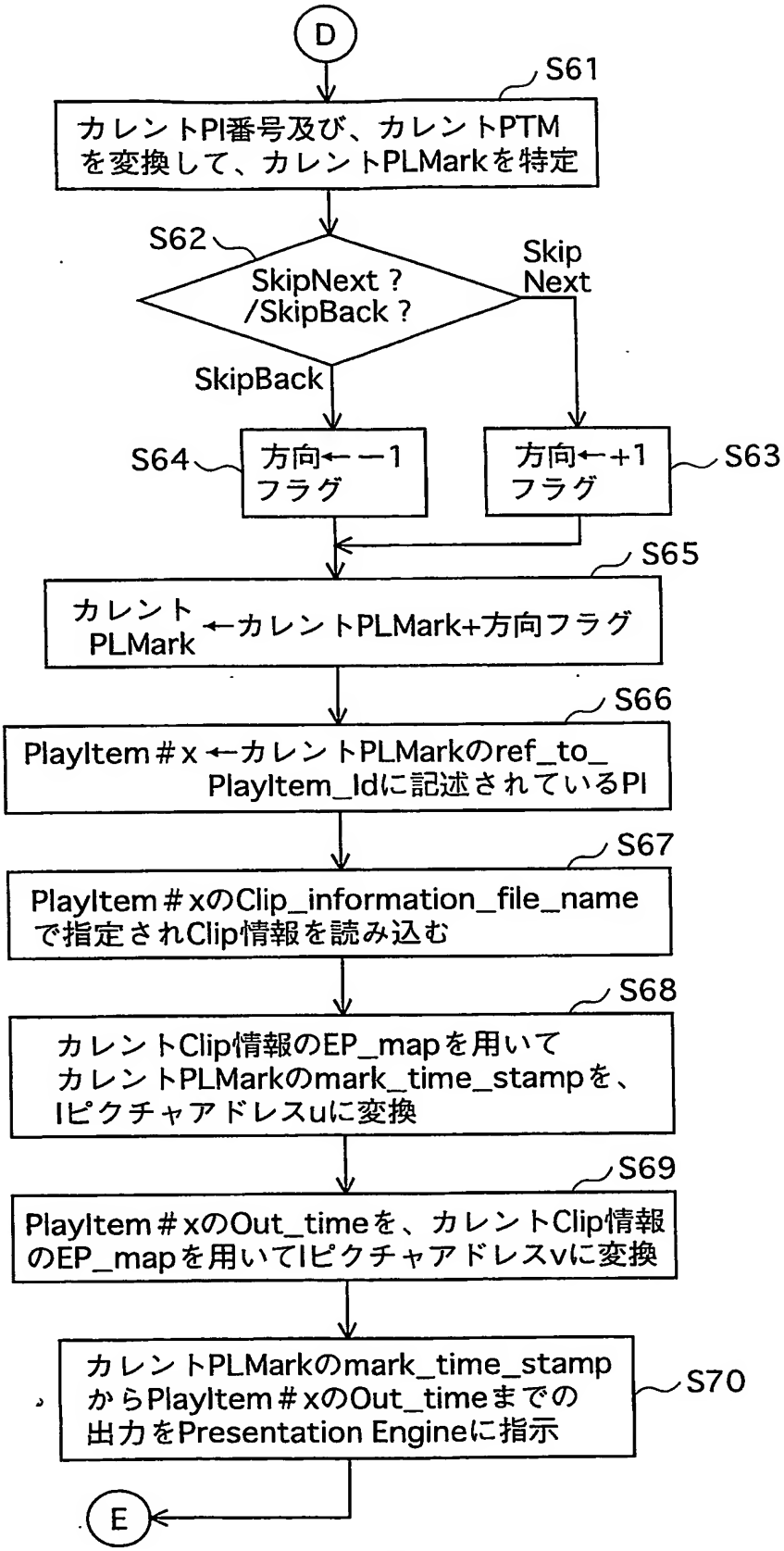

D
S61
カレントPl番号及び、カレントPTM
を変換して、カレントPLMarkを特定
S62
SkipNext ?
/SkipBack ?
Skip
Next
SkipBack
S64
方向←−1
フラグ
方向←+1
フラグ
S63
S65
カレント
PLMark
←カレントPLMark+方向フラグ
S66
PlayItem＃x←カレントPLMarkのref_to_
PlayItem_Idに記述されているPl
S67
PlayItem＃xのClip_information_file_name
で指定されClip情報を読み込む
S68
カレントClip情報のEP_mapを用いて
カレントPLMarkのmark_time_stampを、
Iピクチャアドレスuに変換
S69
PlayItem＃xのOut_timeを、カレントClip情報
のEP_mapを用いてIピクチャアドレスvに変換
カレントPLMarkのmark_time_stamp
からPlayItem＃xのOut_timeまでの
出力をPresentation Engineに指示
S70
E

図36

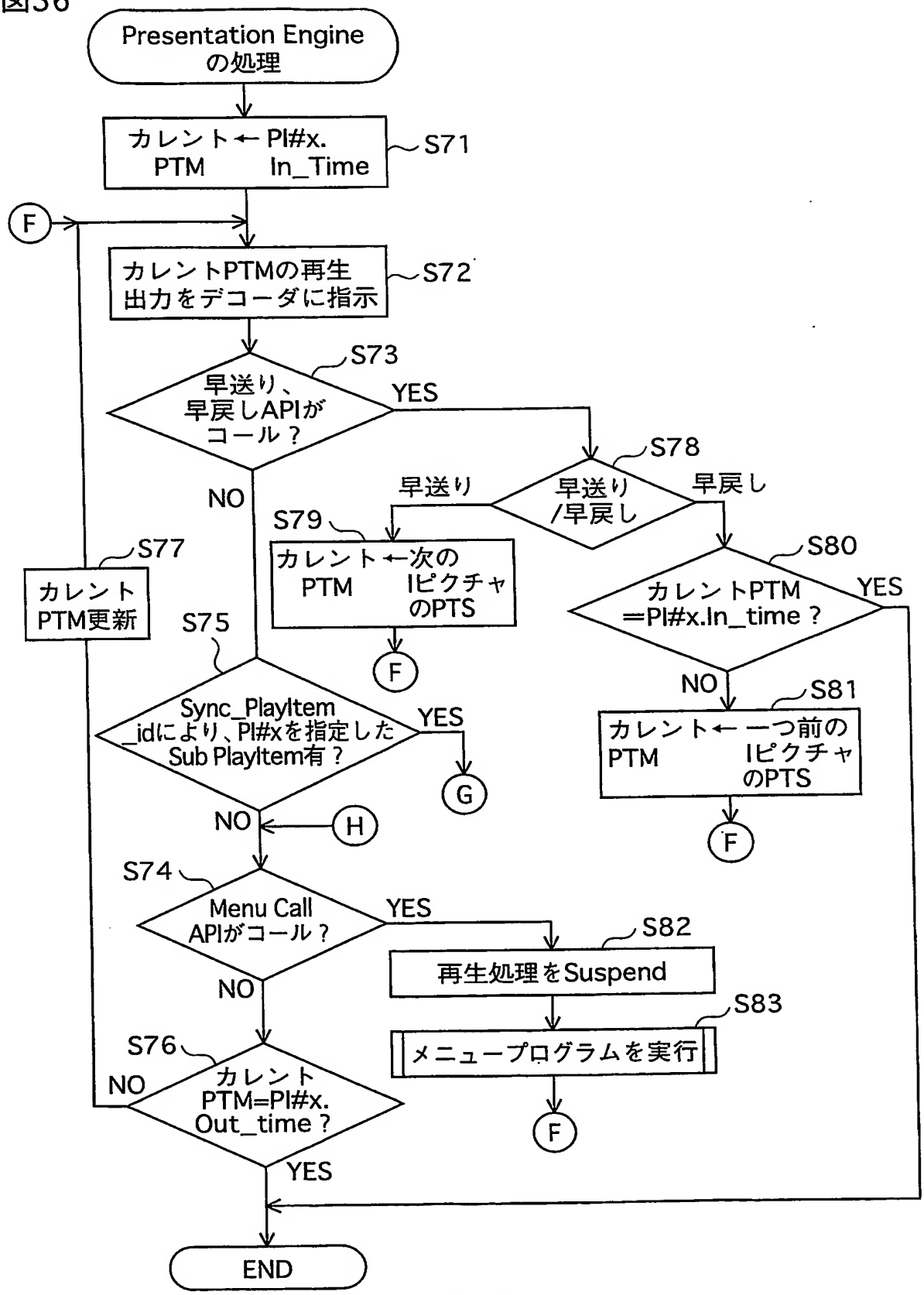

Presentation Engine の処理
カレント← PI#x. PTM In_Time
S71
F
カレントPTMの再生 出力をデコーダに指示
S72
S73
早送り、早戻しAPIがコール？
YES
NO
S78
早送り/早戻し
早送り
早戻し
S79
カレント←次の PTM Iピクチャ のPTS
S80
カレントPTM =PI#x.In_time？
YES
NO
S81
カレント← 一つ前の PTM Iピクチャ のPTS
S77
カレント PTM更新
S75
Sync_PlayItem _idにより、PI#xを指定した Sub PlayItem有？
YES
G
NO
H
S74
Menu Call APIがコール？
YES
S82
再生処理をSuspend
NO
S83
メニュープログラムを実行
S76
カレント PTM=PI#x. Out_time？
NO
YES
END

図37

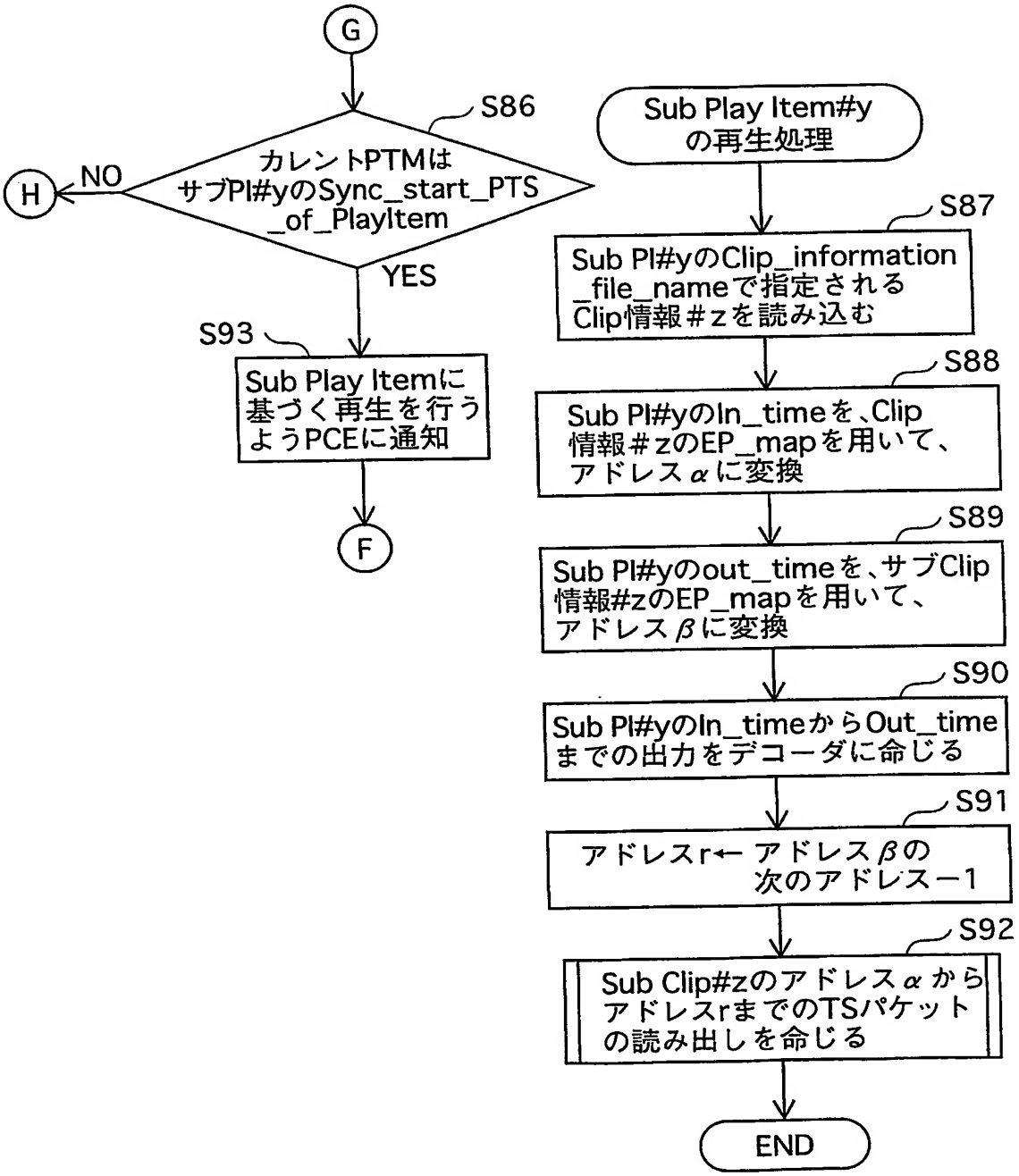

G
S86
カレントPTMは
サブPI#yのSync_start_PTS
_of_PlayItem
NO
H
YES
S93
Sub Play Itemに
基づく再生を行う
ようPCEに通知
F
Sub Play Item#y
の再生処理
S87
Sub PI#yのClip_information
_file_nameで指定される
Clip情報＃zを読み込む
S88
Sub PI#yのIn_timeを、Clip
情報＃zのEP_mapを用いて、
アドレスαに変換
S89
Sub PI#yのout_timeを、サブClip
情報#zのEP_mapを用いて、
アドレスβに変換
S90
Sub PI#yのIn_timeからOut_time
までの出力をデコーダに命じる
S91
アドレスr← アドレスβの
次のアドレス－1
S92
Sub Clip#zのアドレスαから
アドレスrまでのTSパケット
の読み出しを命じる
END

図38

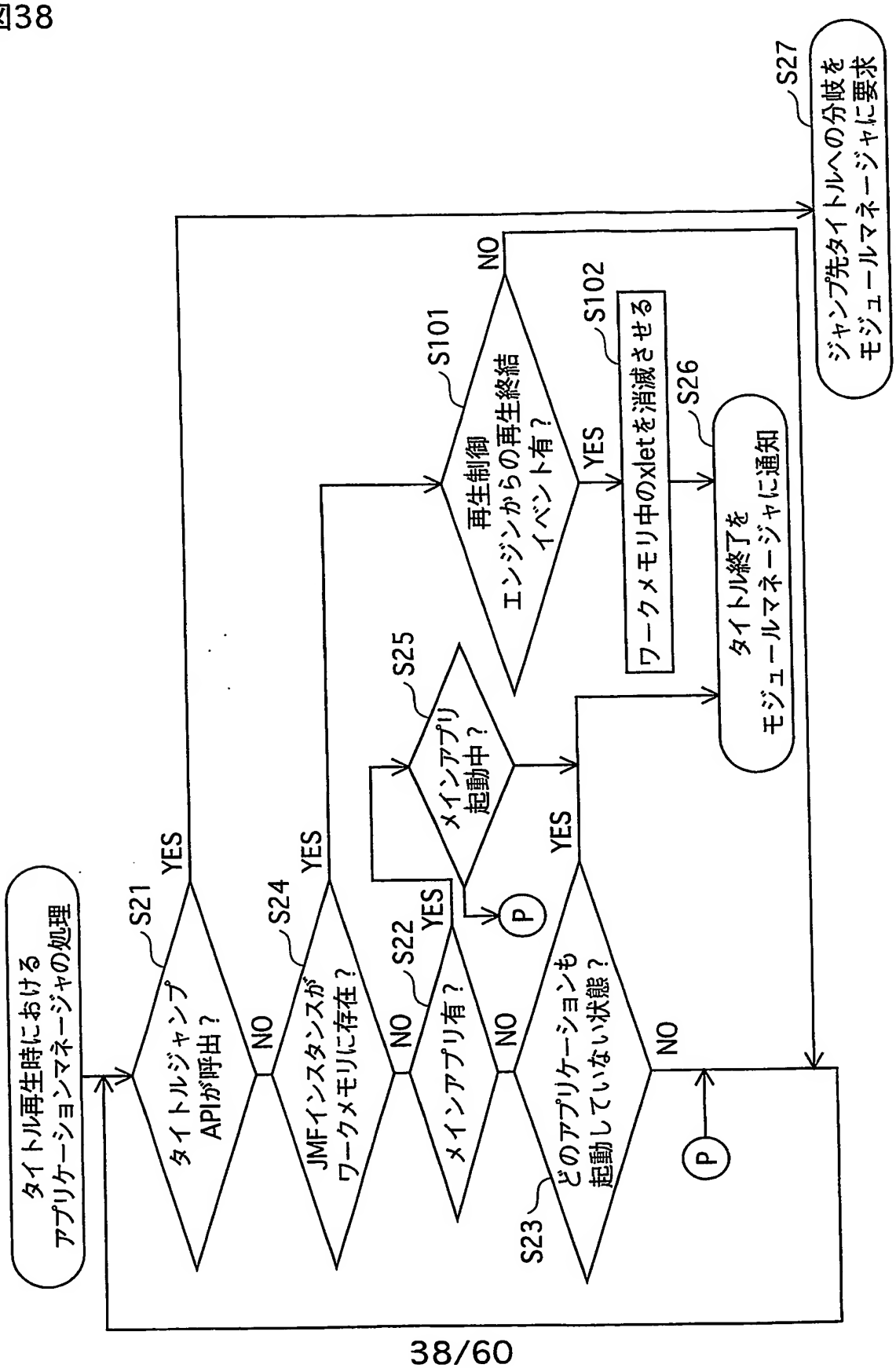

タイトル再生時における
アプリケーションマネージャの処理
S21
タイトルジャンプ
APIが呼出？
YES
NO
S24
JMFインスタンスが
ワークメモリに存在？
YES
NO
S22
メインアプリ有？
YES
NO
S23
どのアプリケーションも
起動していない状態？
YES
NO
S25
メインアプリ
起動中？
P
P
S101
再生制御
エンジンからの再生終結
イベント有？
NO
YES
S102
ワークメモリ中のxletを消滅させる
S26
タイトル終了を
モジュールマネージャに通知
S27
ジャンプ先タイトルへの分岐を
モジュールマネージャに要求

図39

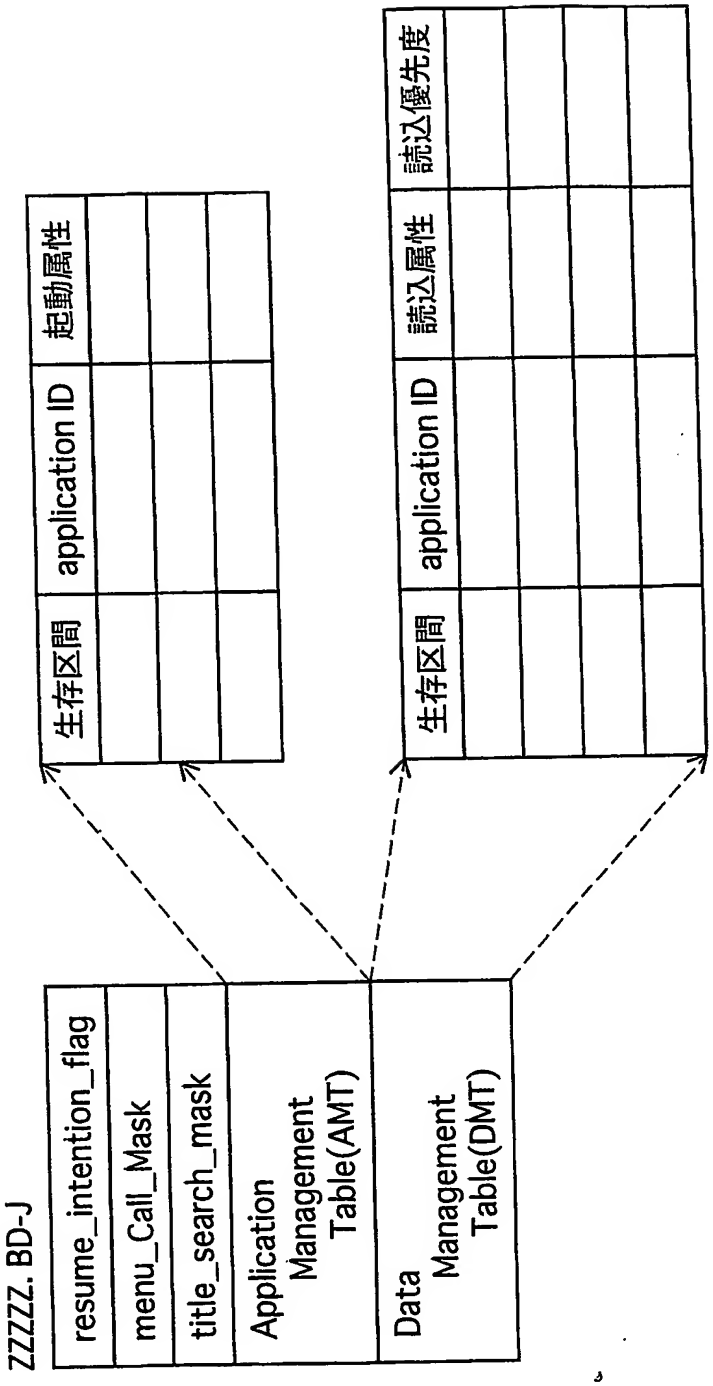

ZZZZZ. BD-J
resume_intention_flag
menu_Call_Mask
title_search_mask
Application Management Table(AMT)
Data Management Table(DMT)
生存区間
application ID
起動属性
生存区間
application ID
読込属性
読込優先度

図40

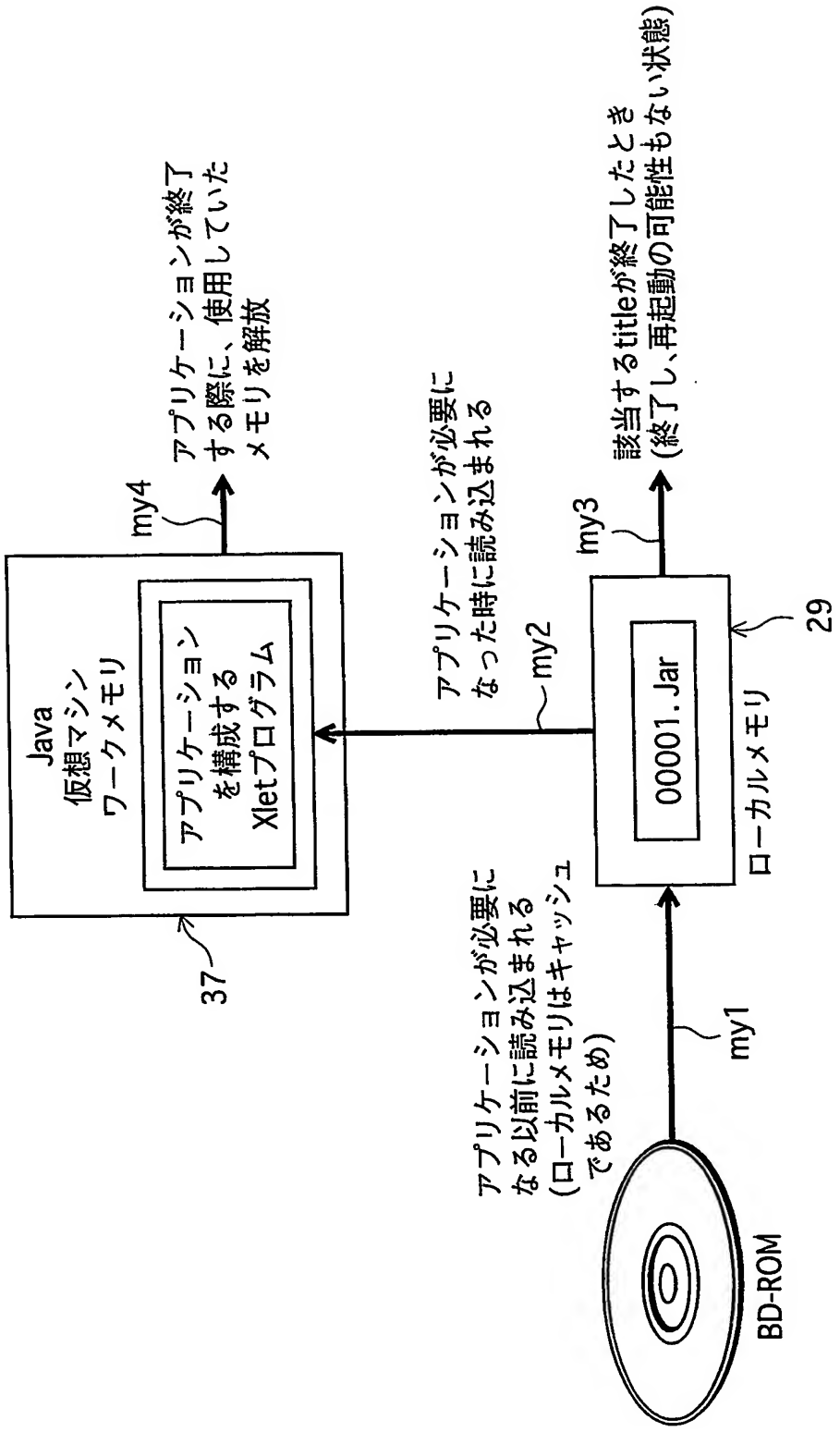

BD-ROM
my1
アプリケーションが必要に
なる以前に読み込まれる
(ローカルメモリはキャッシュ
であるため)
00001.Jar
ローカルメモリ
29
my2
アプリケーションが必要に
なった時に読み込まれる
my3
該当するtitleが終了したとき
(終了し、再起動の可能性もない状態)
Java
仮想マシン
ワークメモリ
アプリケーション
を構成する
Xletプログラム
my4
アプリケーションが終了
する際に、使用していた
メモリを解放
37

図41

(a)

title #1のデータ管理テーブル

生存区間	アプリケーションID	読込属性	読込優先度
title #1	application #1		
title #1	application #2		

title #2のデータ管理テーブル

生存区間	アプリケーションID	読込属性	読込優先度
title #1	application #1		
title #2	application #2		

(b)

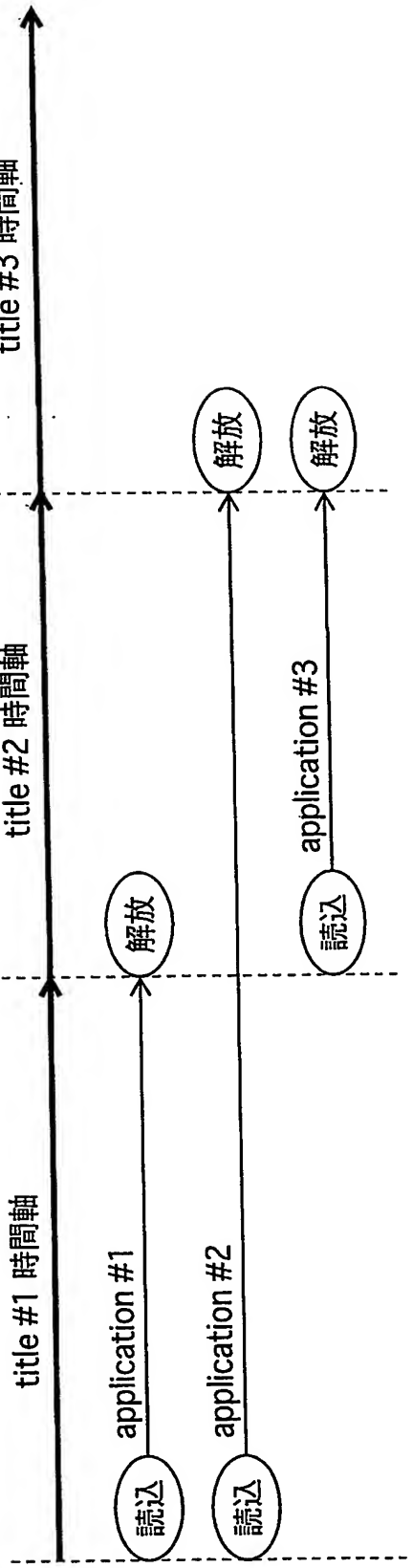


図42

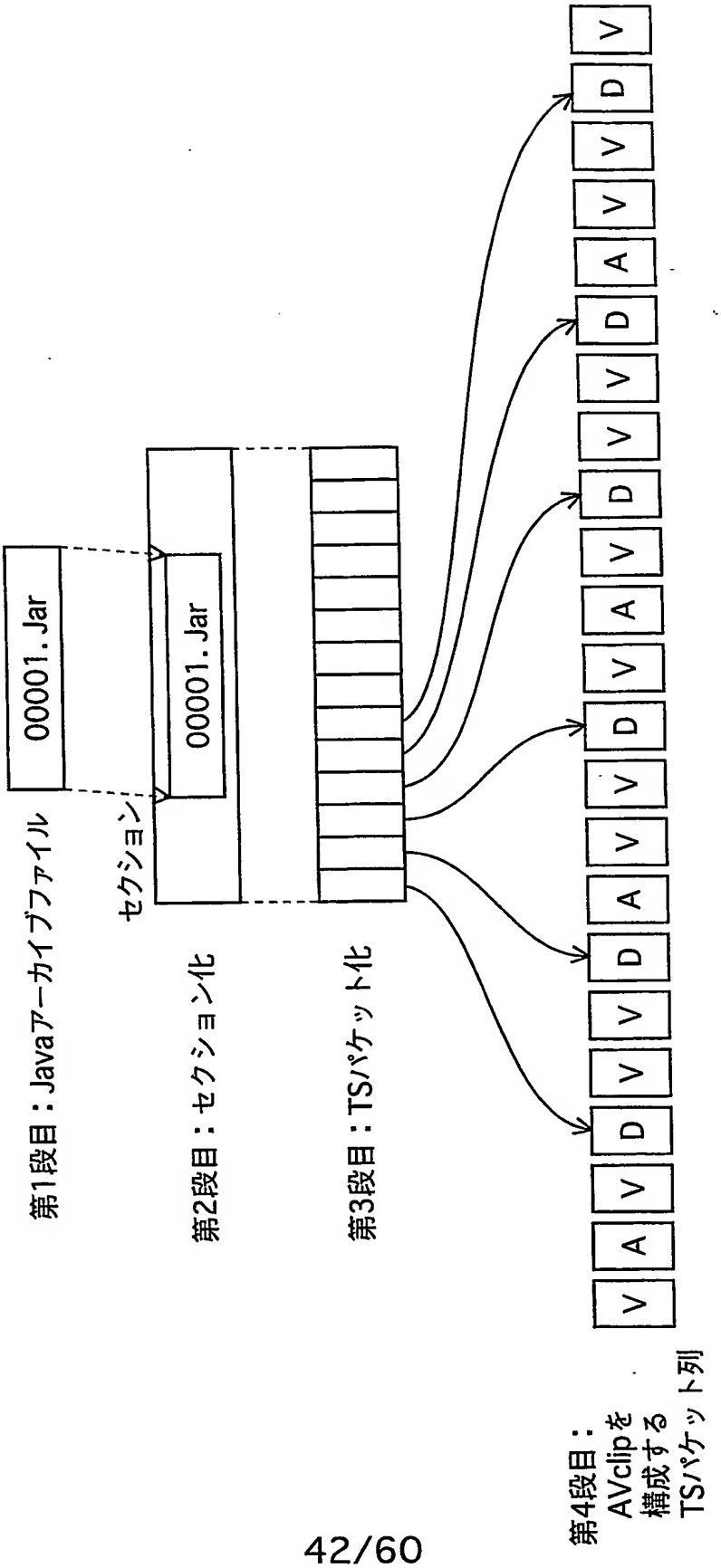

00001.Jar
00001.Jar
セクション
第1段目：Javaアーカイブファイル
第2段目：セクション化
第3段目：TSパケット化
第4段目：
AVclipを
構成する
TSパケット列
V
A
V
D
V
V
D
A
V
V
D
A
V
V
D
A
V
V
D
A
V
V
D
A
V
V
D
V
V
V
A
V
V
D

図43

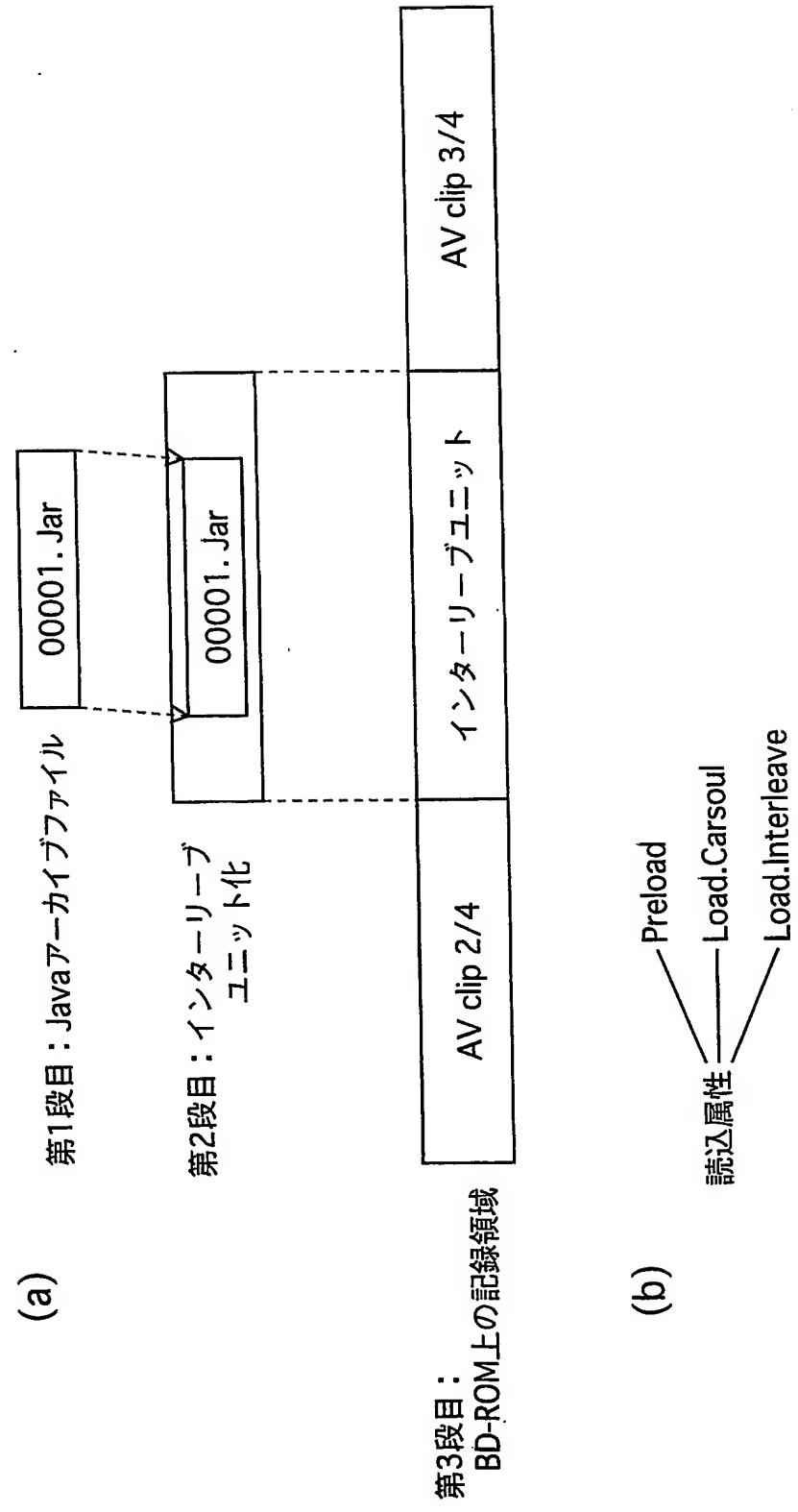

(a)
第1段目：Javaアーカイブファイル
00001.Jar
第2段目：インターリーブユニット化
00001.Jar
インターリーブユニット
AV clip 3/4
AV clip 2/4
第3段目：
BD-ROM上の記録領域
(b)
読込属性
Preload
Load.Carsoul
Load.Interleave

図44

(a) title #1のデータ管理テーブル

生存区間	アプリケーションID	読込属性	読込優先度
title #1	application #1	Preload	mandatory
title #1:chapter #1-#2	application #2	Load	optional
title #1:chapter #4-#5	application #3	Load	optional

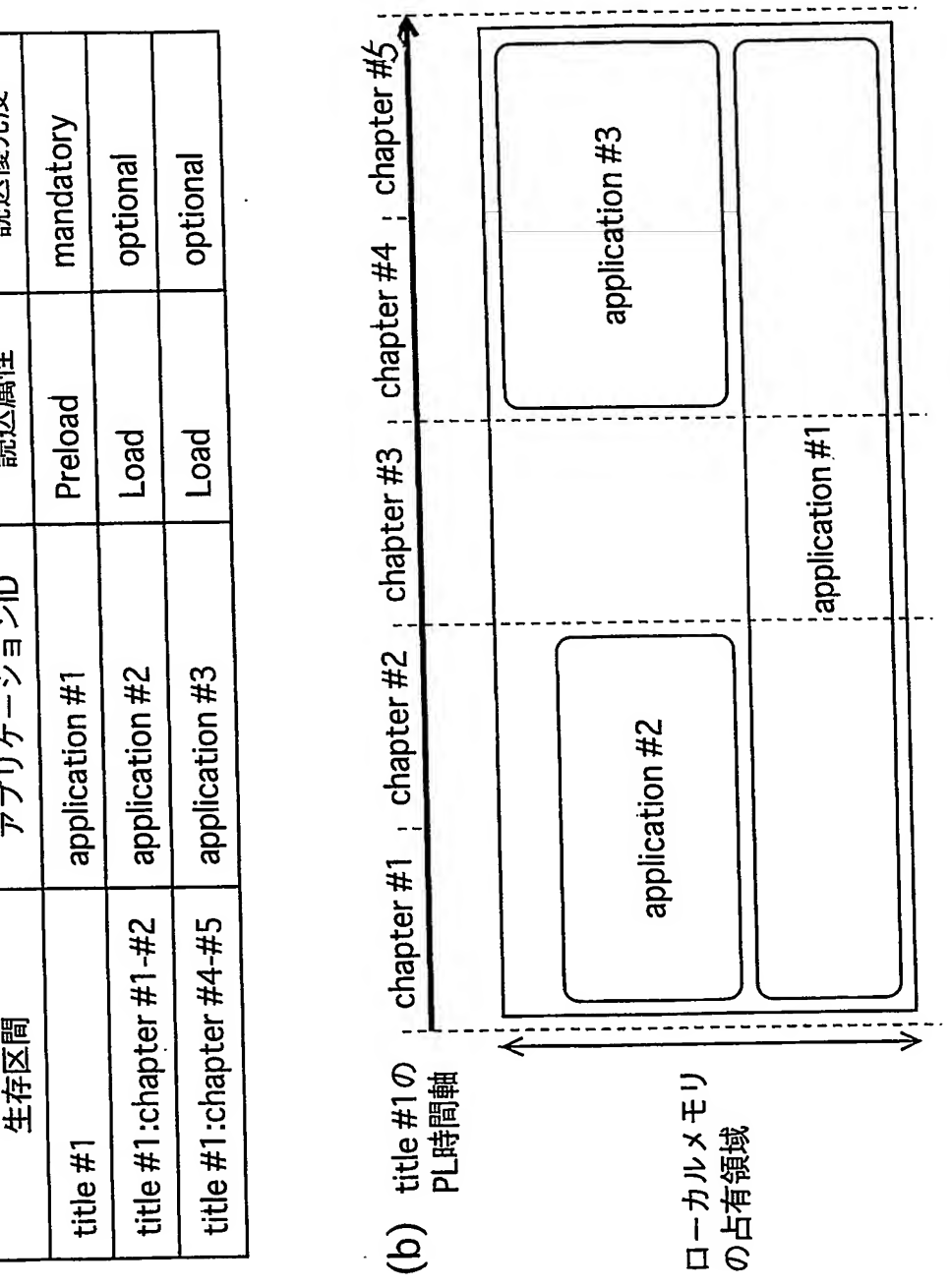
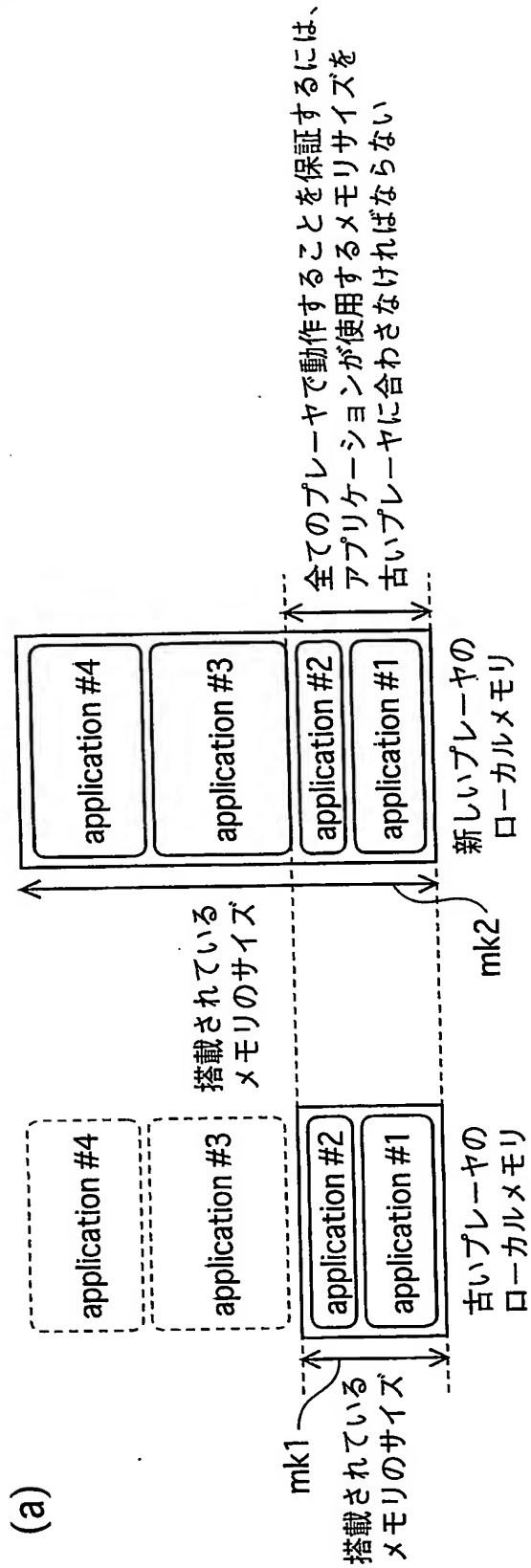


図45



(b) title #1のデータ管理テーブル

タイトル番号	アプリケーションID	読込属性	読込優先度
title #1	application #1	Preload	mandatory
title #1	application #2	Preload	mandatory
title #1	application #3	Preload	optional
title #1	application #4	Preload	optional

図46

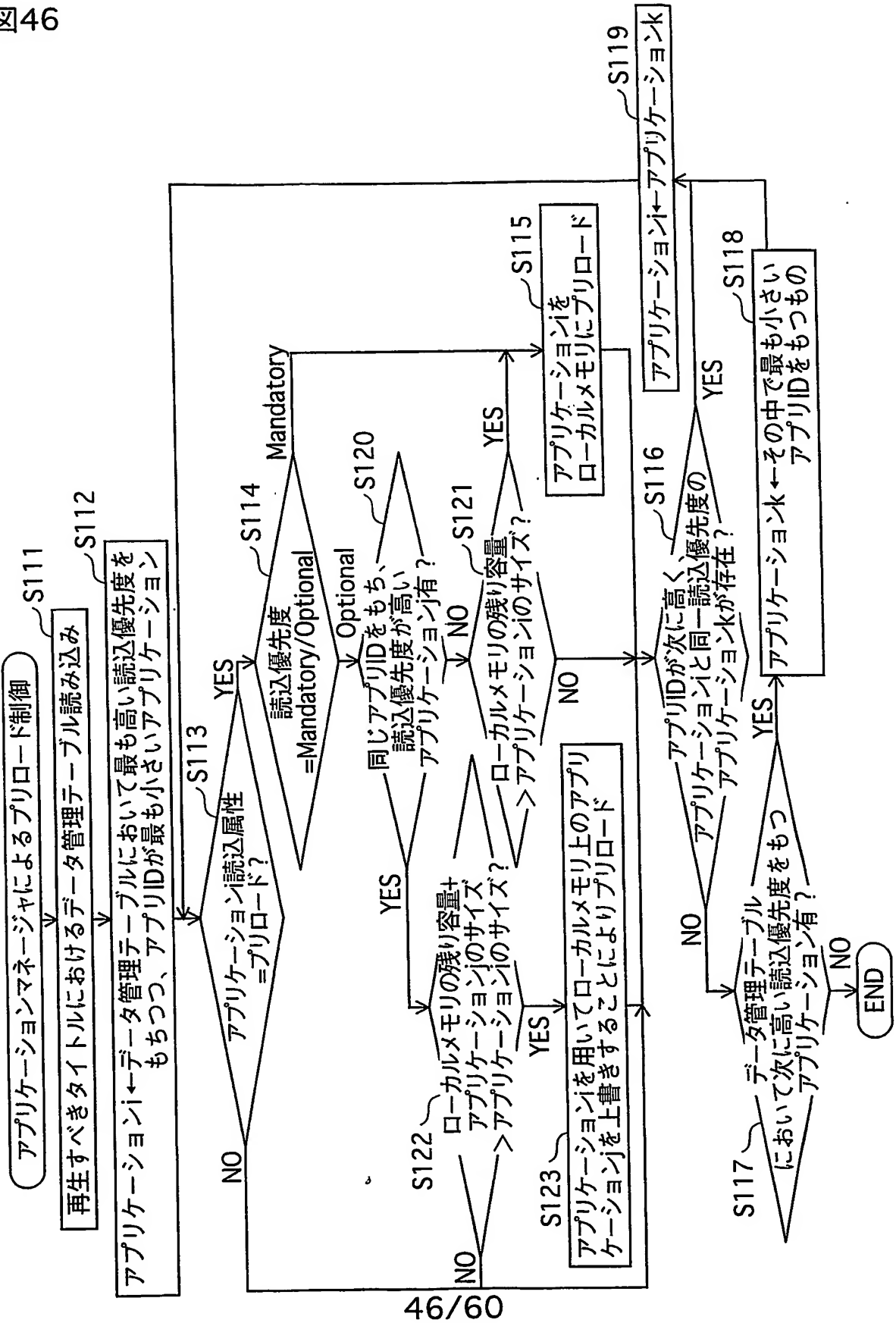

アプリケーションマネージャによるプリロード制御
S111
再生すべきタイトルにおけるデータ管理テーブル読み込み
S112
アプリケーションi←データ管理テーブルにおいて最も高い読込優先度を
もちつつ、アプリIDが最も小さいアプリケーション
S113
アプリケーションi読込属性
=プリロード?
YES
S114
読込優先度
=Mandatory/Optional
Mandatory
Optional
S120
同じアプリIDをもち、
読込優先度が高い
アプリケーションi有?
YES
S121
ローカルメモリの残り容量
>アプリケーションiのサイズ?
YES
S115
アプリケーションiを
ローカルメモリにプリロード
NO
S122
ローカルメモリの残り容量+
アプリケーションiのサイズ
>アプリケーションiのサイズ?
NO
S123
アプリケーションiを用いてローカルメモリ上のアプリ
ケーションjを上書きすることによりプリロード
S116
アプリIDが次に高く、
アプリケーションiと同一読込優先度の
アプリケーションkが存在?
YES
S118
アプリケーションk←その中で最も小さい
アプリIDをもつもの
S119
アプリケーションi←アプリケーションk
NO
S117
データ管理テーブル
において次に高い読込優先度をもつ
アプリケーション有?
YES
NO
END

図47

(a) title #1のデータ管理テーブル

゛生存区間	アプリケーションID	読込属性	読込優先度
title #1	application #1	Preload	mandatory
title #1	application #1	Preload	optional:high
title #1	application #1	Preload	optional:low

(b) ローカルメモリの占有領域

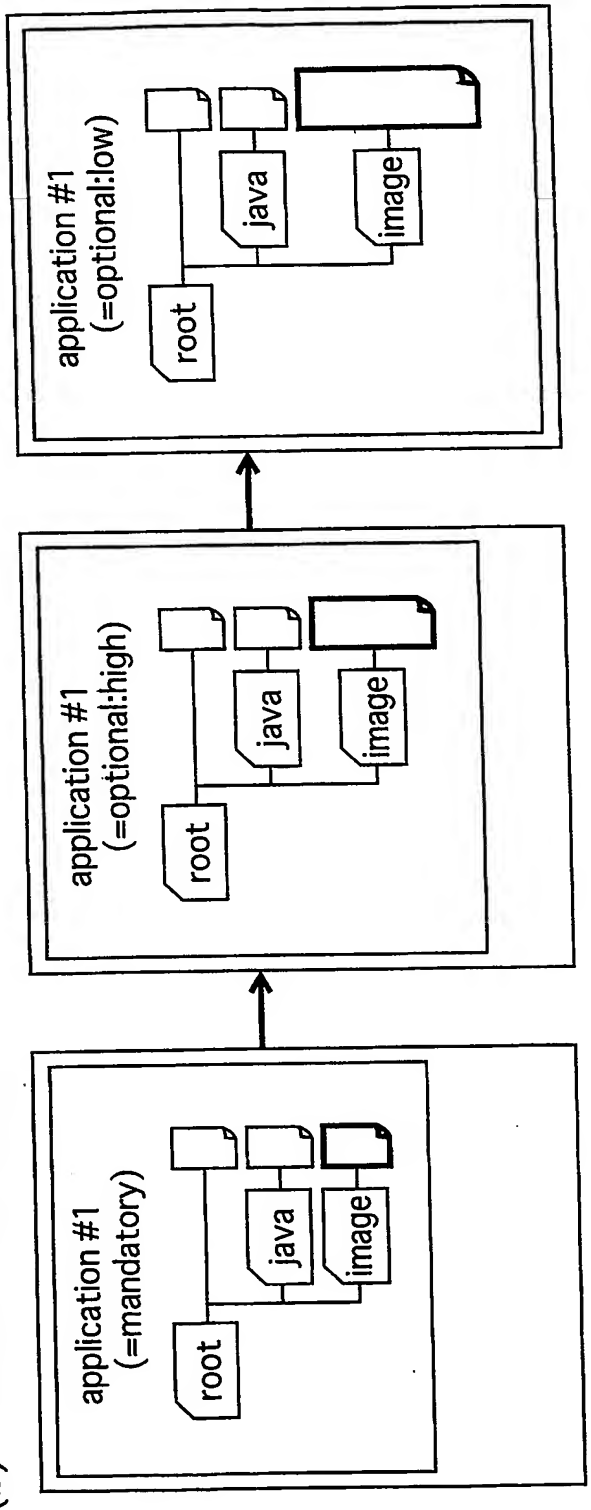


図48

(a) title #1のデータ管理テーブル

生存区間	アプリケーションID	読込属性	読込優先度
title #1	application #1	Preload	mandatory
title #1:chapter #1-#2	application #2	Load	mandatory
title #1:chapter #4-#5	application #3	Load	mandatory
title #1	application #3	Preload	optional

図49

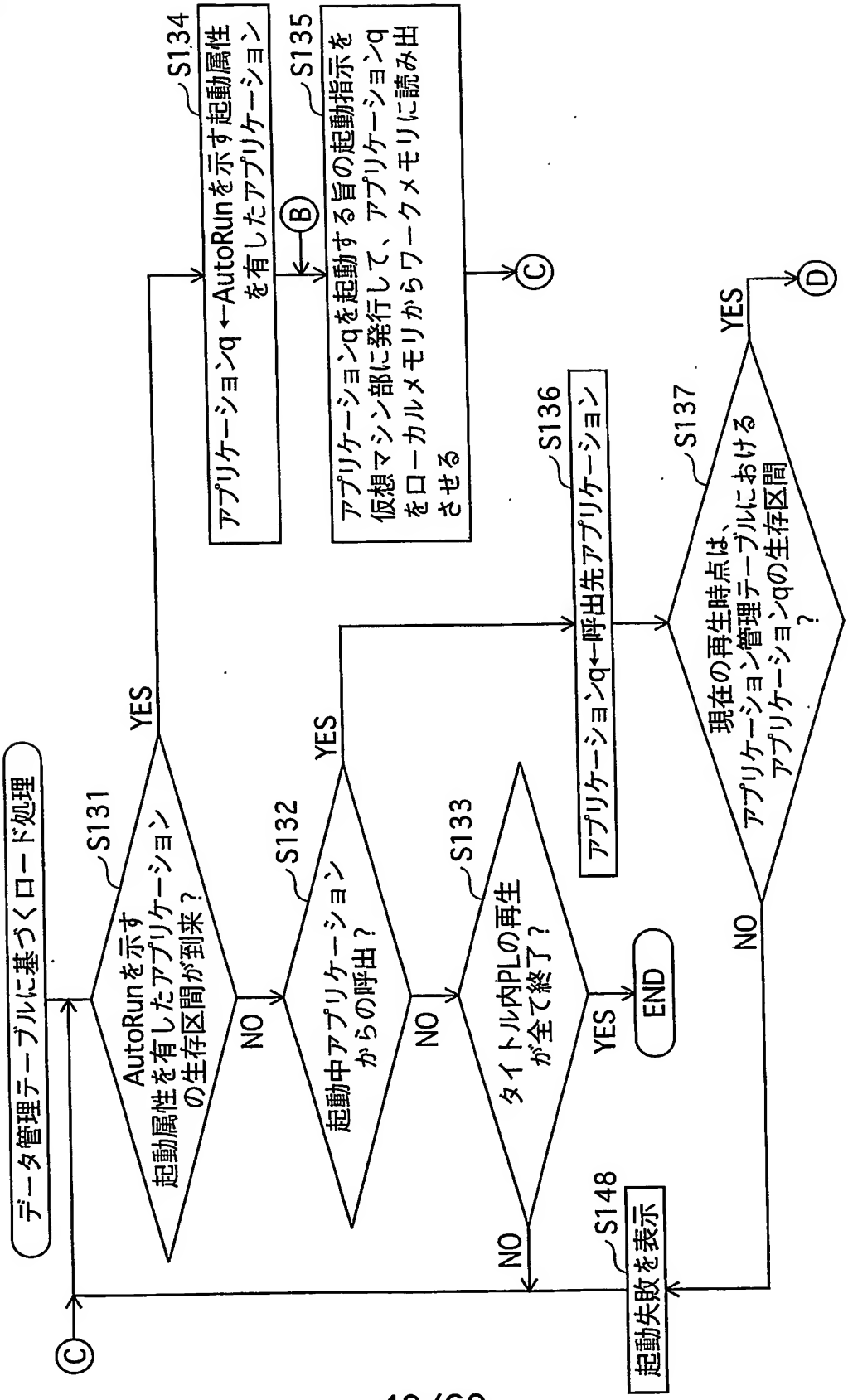

データ管理テーブルに基づくロード処理
S131
AutoRunを示す起動属性を有したアプリケーションの生存区間が到来？
YES
NO
S132
起動中アプリケーションからの呼出？
YES
NO
S133
タイトル内PLの再生が全て終了？
YES
NO
END
S134
アプリケーションq←AutoRunを示す起動属性を有したアプリケーション
S135
アプリケーションqを起動する旨の起動指示を仮想マシン部に発行して、アプリケーションqをローカルメモリからワークメモリに読み出させる
S136
アプリケーションq←呼出先アプリケーション
S137
現在の再生時点は、アプリケーション管理テーブルにおけるアプリケーションqの生存区間？
YES
NO
S148
起動失敗を表示
Ⓑ
Ⓒ
Ⓓ

図50

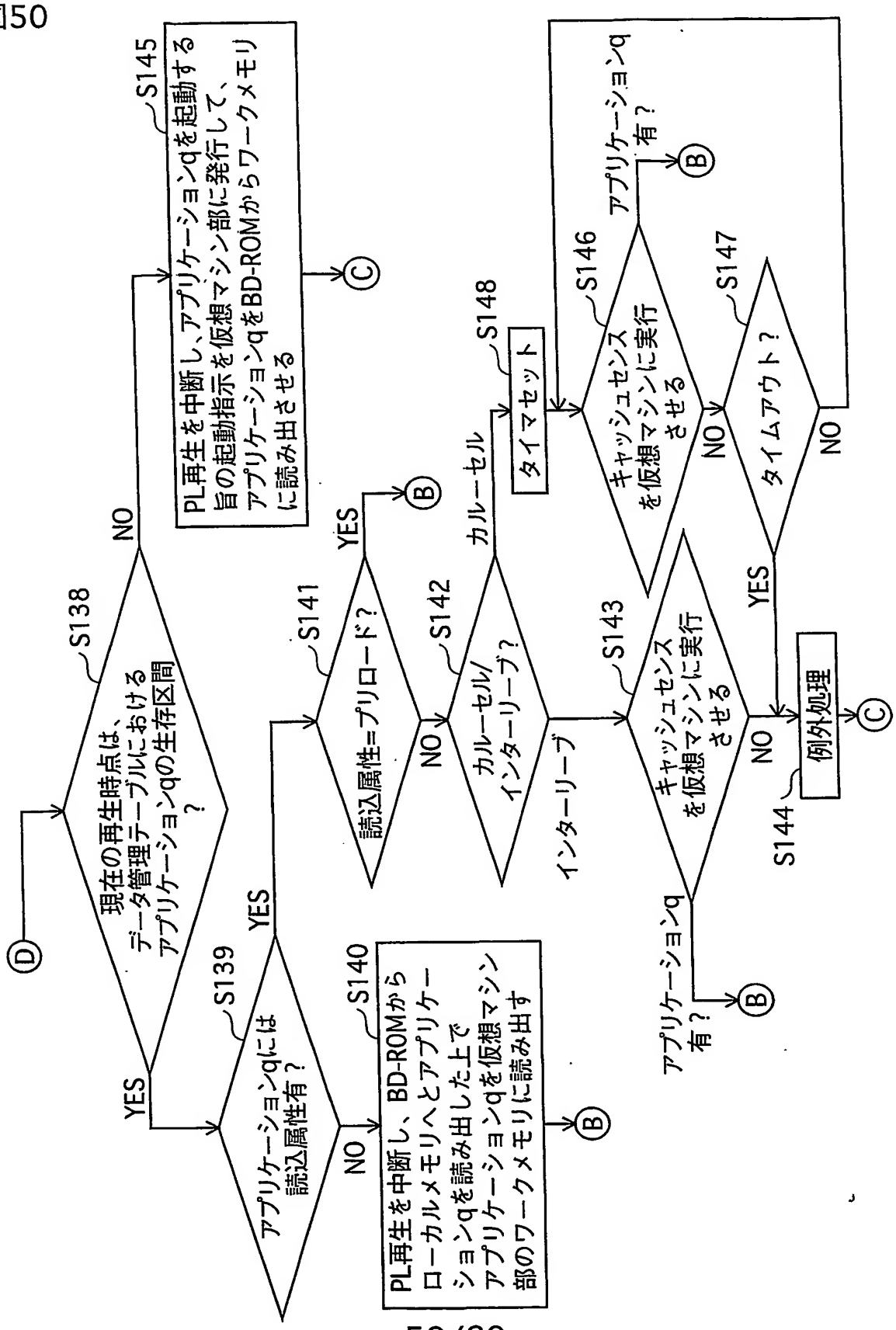

S138
現在の再生時点は、データ管理テーブルにおけるアプリケーションqの生存区間？
NO
YES
S139
アプリケーションqには読込属性有？
YES
NO
S140
PL再生を中断し、BD-ROMからローカルメモリへとアプリケーションqを読み出した上でアプリケーションqを仮想マシン部のワークメモリに読み出す
B
S141
読込属性＝プリロード？
YES
NO
B
S142
カルーセル/インターリーブ？
カルーセル
インターリーブ
S143
キャッシュセンスを仮想マシンに実行させる
アプリケーションq有？
YES
NO
B
S144
例外処理
C
S145
PL再生を中断し、アプリケーションqを起動する旨の起動指示を仮想マシン部に発行して、アプリケーションqをBD-ROMからワークメモリに読み出させる
C
S146
タイマセット
キャッシュセンスを仮想マシンに実行させる
アプリケーションq有？
YES
NO
B
S147
タイムアウト？
YES
NO
S148
タイマセット
D

図51

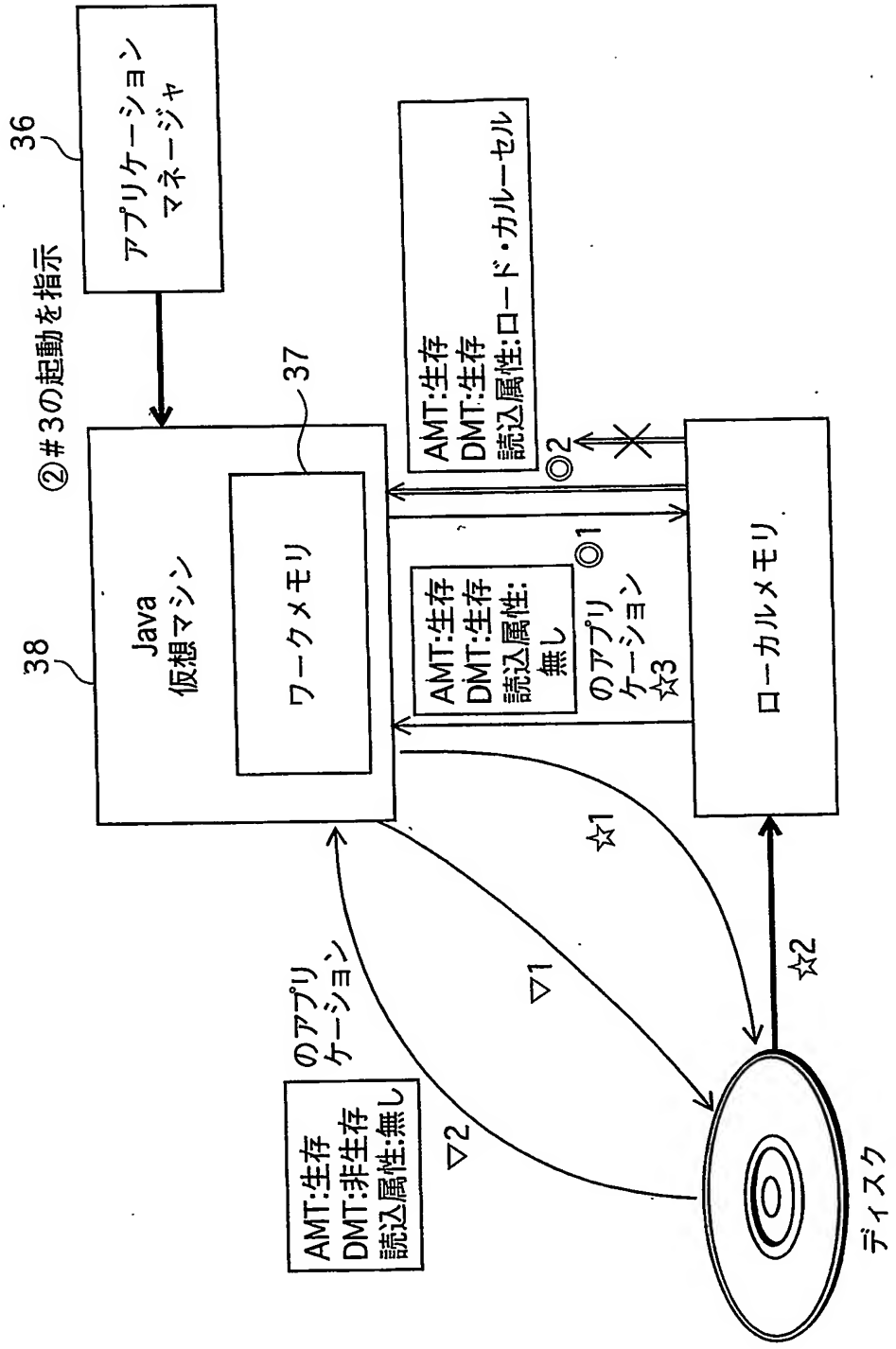

アプリケーション
マネージャ
36
②#3の起動を指示
37
Java
仮想マシン
38
ワークメモリ
AMT:生存
DMT:生存
読込属性:ロード・カルーセル
◎2
AMT:生存
DMT:生存
読込属性:
無し
のアプリ
ケーション
☆3
◎1
ローカルメモリ
☆1
▽1
☆2
▽2
のアプリ
ケーション
AMT:生存
DMT:非生存
読込属性:無し
ディスク

図52

(a) ZZZZZ. BD-J

resume_intention_flag
menu_Call_Mask
title_search_mask
Application Management Table(AMT)
Data Management Table(DMT)
PlayList Management Table(PLMT)

(b) PL管理テーブル

プレイリストID	再生属性
ーー	Auto Play
ーー	ーー

(c)

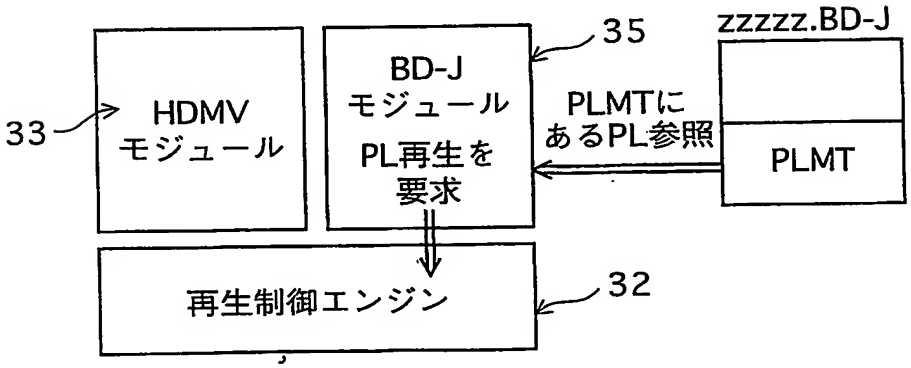


図53

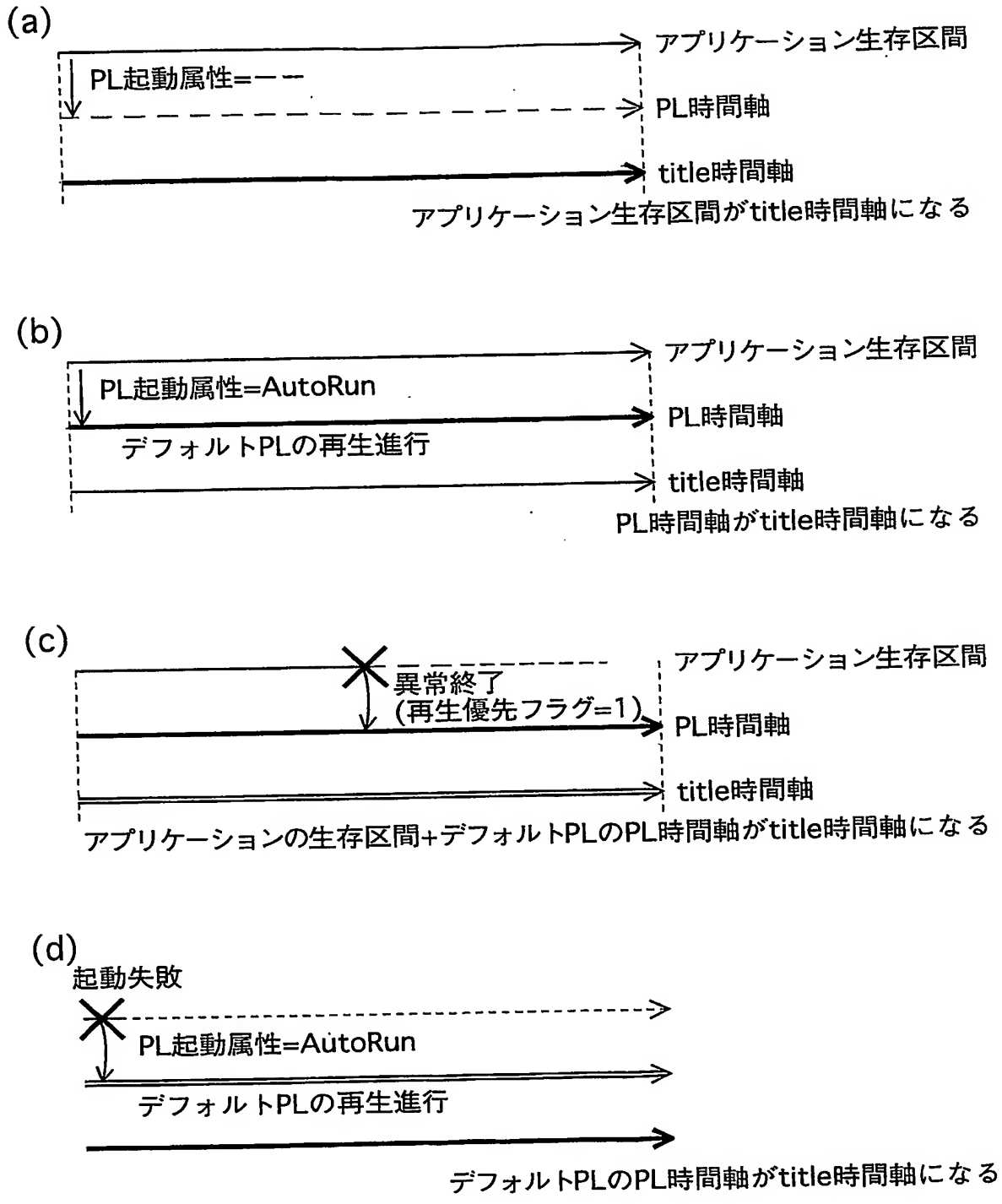

(a)
アプリケーション生存区間
PL起動属性=－－－
PL時間軸
title時間軸
アプリケーション生存区間がtitle時間軸になる
(b)
アプリケーション生存区間
PL起動属性=AutoRun
PL時間軸
デフォルトPLの再生進行
title時間軸
PL時間軸がtitle時間軸になる
(c)
アプリケーション生存区間
異常終了
(再生優先フラグ=1)
PL時間軸
title時間軸
アプリケーションの生存区間+デフォルトPLのPL時間軸がtitle時間軸になる
(d)
起動失敗
PL起動属性=AutoRun
デフォルトPLの再生進行
デフォルトPLのPL時間軸がtitle時間軸になる

図54

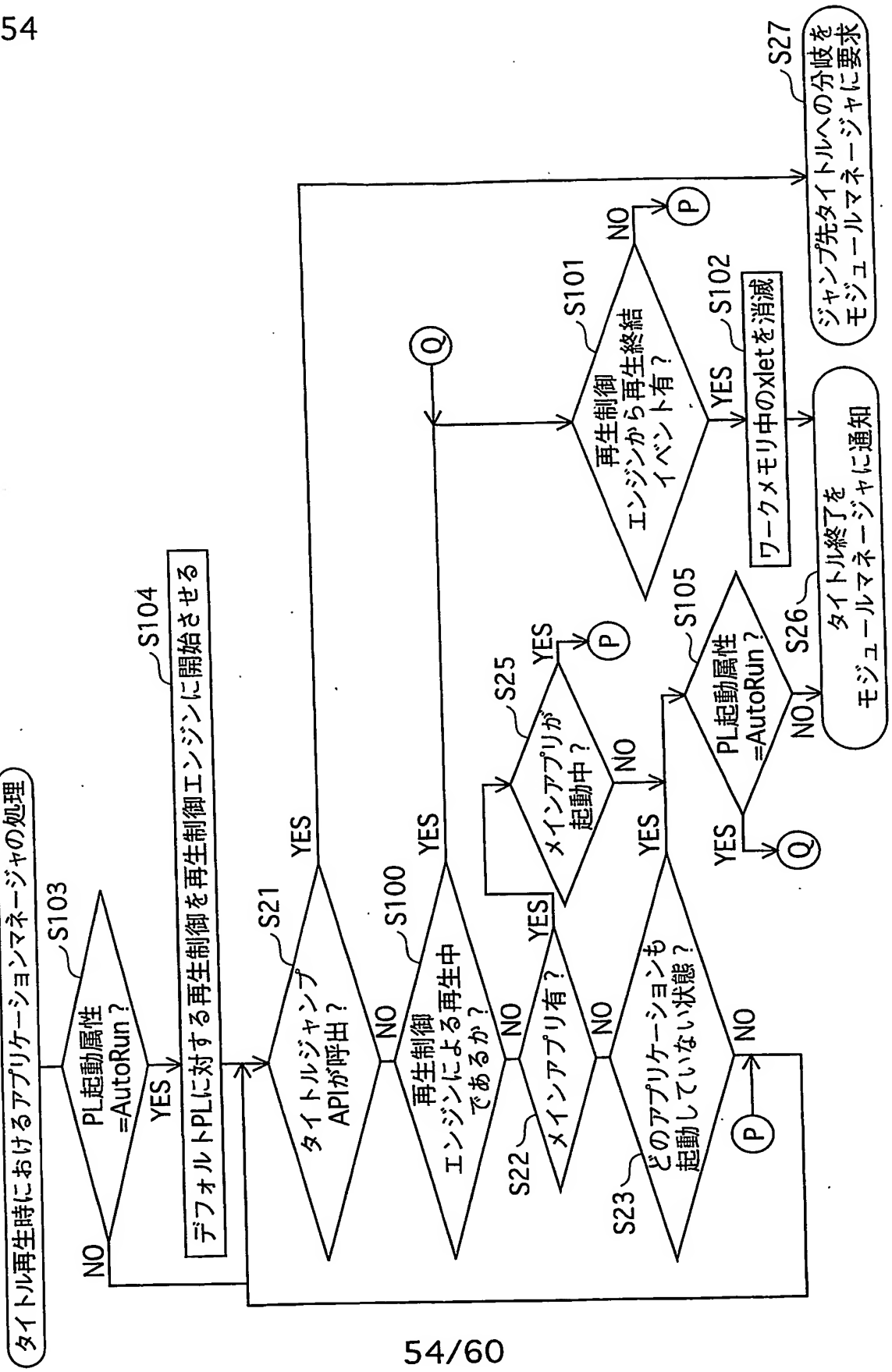

タイトル再生時におけるアプリケーションマネージャの処理
S103
PL起動属性 =AutoRun？
YES
NO
S104
デフォルトPLに対する再生制御を再生制御エンジンに開始させる
S21
タイトルジャンプ APIが呼出？
YES
NO
S100
再生制御 エンジンによる再生中 であるか？
YES
NO
S22
メインアプリ有？
YES
NO
S23
どのアプリケーションも 起動していない状態？
YES
NO
S25
メインアプリが 起動中？
YES
NO
S105
PL起動属性 =AutoRun？
YES
NO
S101
再生制御 エンジンから再生終結 イベント有？
YES
NO
S102
ワークメモリ中のxletを消滅
S26
タイトル終了を モジュールマネージャに通知
S27
ジャンプ先タイトルへの分岐を モジュールマネージャに要求
P
Q

図55

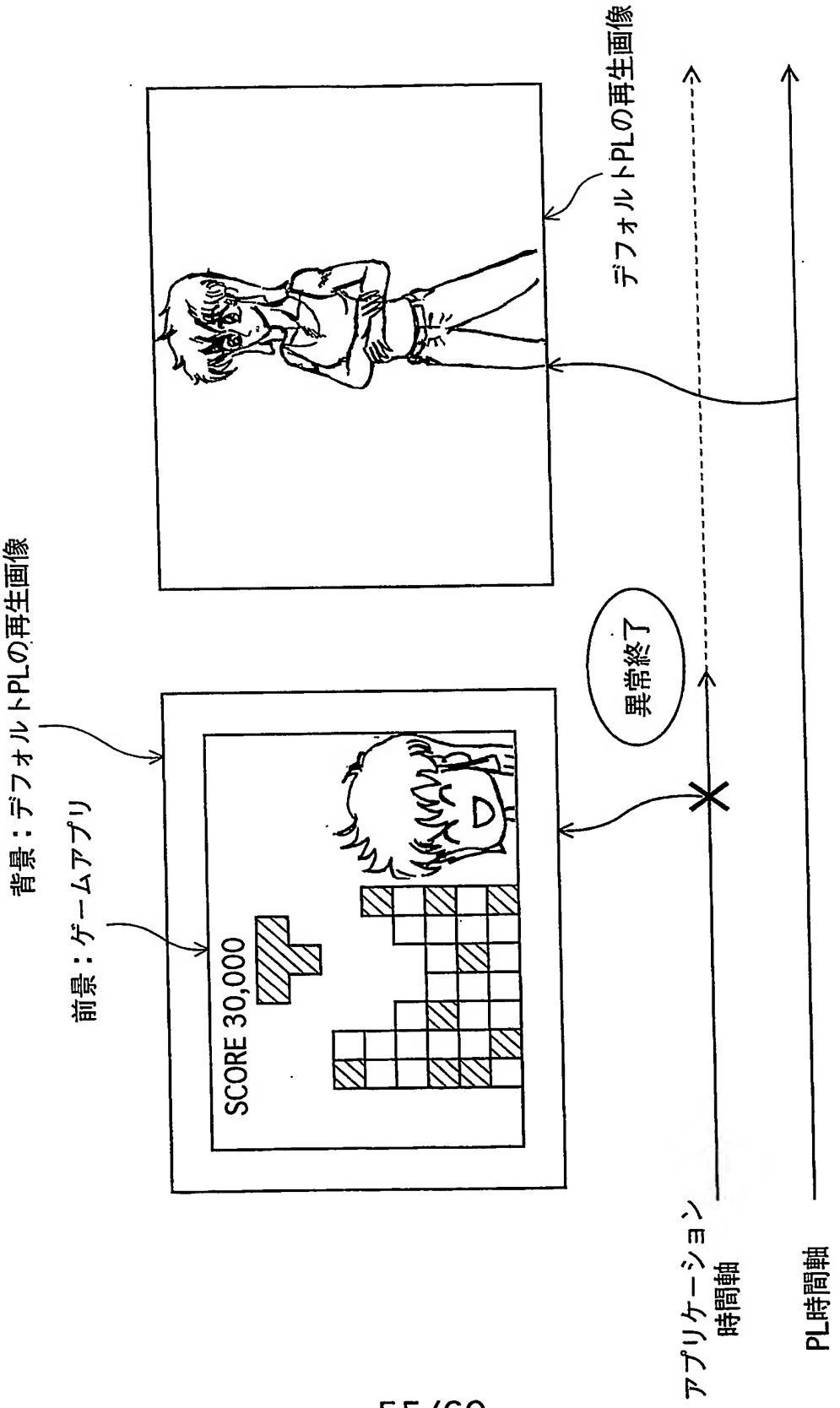

デフォルトPLの再生画像
異常終了
背景：デフォルトPLの再生画像
前景：ゲームアプリ
SCORE 30,000
アプリケーション
時間軸
PL時間軸

図56

(a)

アプリケーション・データ管理テーブル

生存区間	アプリケーションID	起動属性	読込優先度
title #1	application #1	AutoRun	
title #1:chapter #1-#3	application #2	Ready	
title #1	application #3	ーー	
title #1:chapter #2-#4	application #4	Ready	

起動属性+読込属性

(b)

起動属性	AVストリーム再生前のプリロード	自動起動/呼出起動	ローカルメモリへのロード	生存/非生存
AutoRun	○	自動	×	生存
AutoRun	×	自動	○	生存
READY	○	呼出	×	生存
READY	×	呼出	○	生存
無指定	×	呼出	×	生存

図57

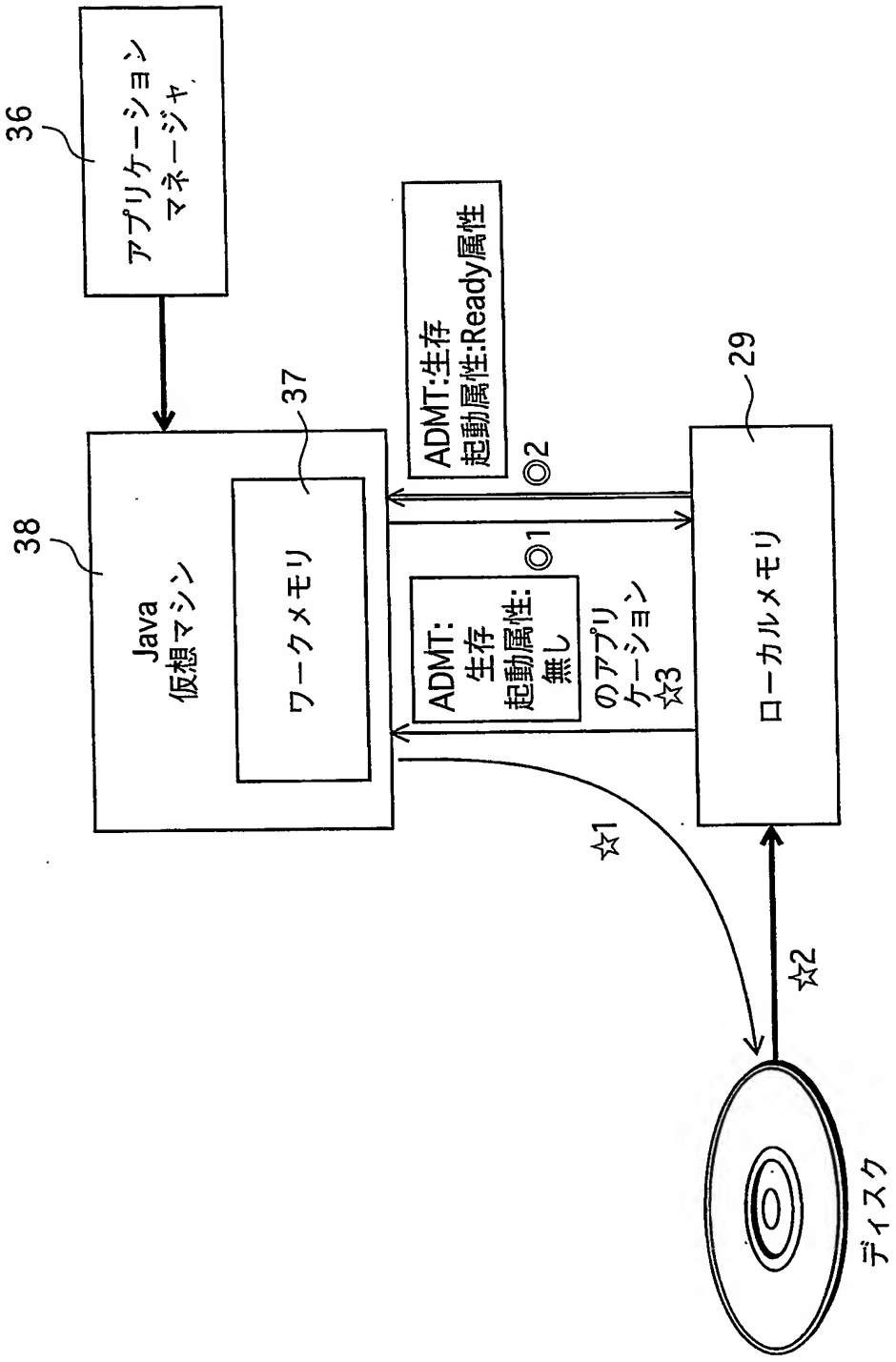

アプリケーション
マネージャ
36
37
38
29
Java
仮想マシン
ワークメモリ
ADMT:生存
起動属性:Ready属性
◎2
◎1
ADMT:
生存
起動属性:
無し
のアプリ
ケーション
☆3
ローカルメモリ
☆1
☆2
ディスク

図58

(a) データ管理情報

生存区間	application ID	読込属性	読込優先度
title #1	application #1	Preload	mandatory
title #1	application #2	Preload	option, 255
title #1	application #3	Preload	option, 128

(b)

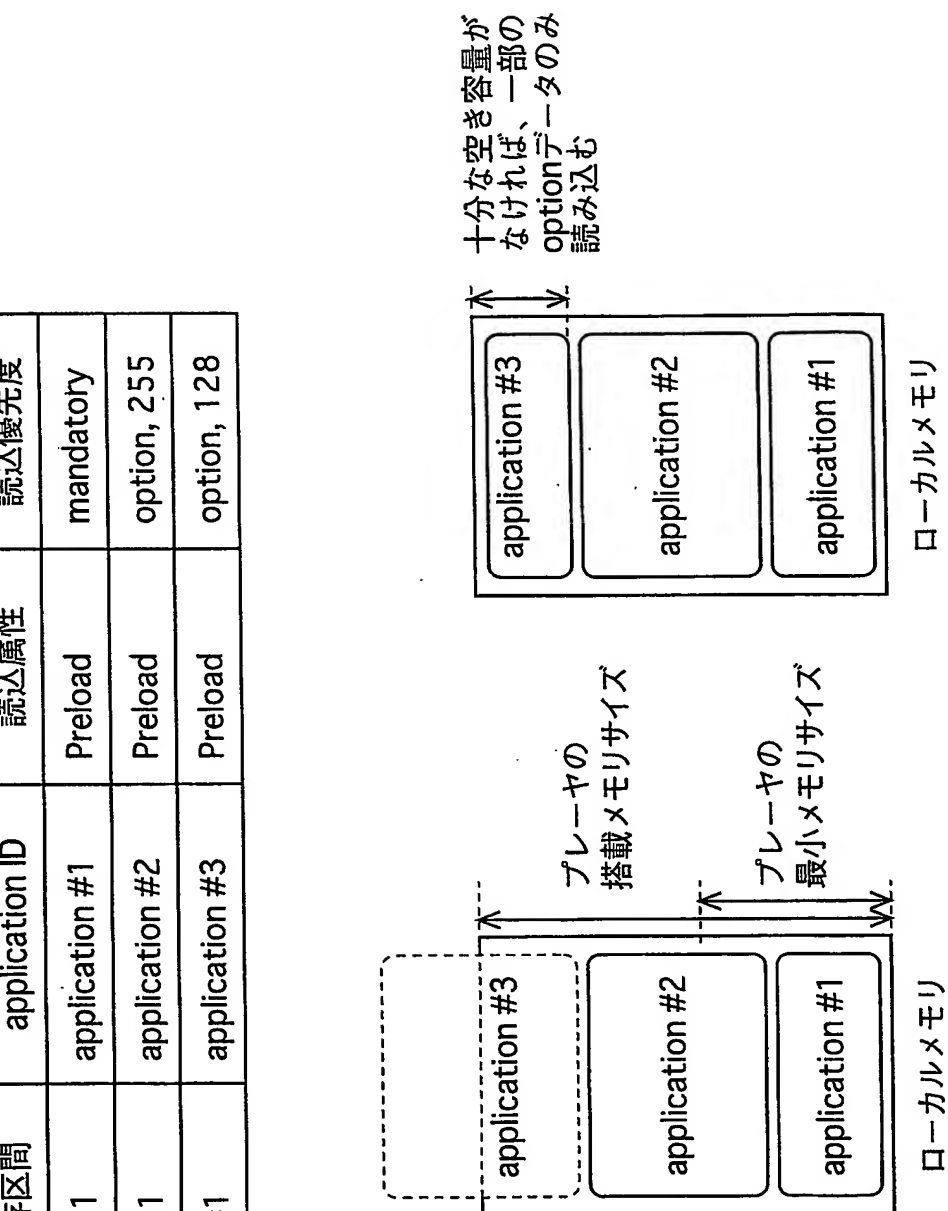
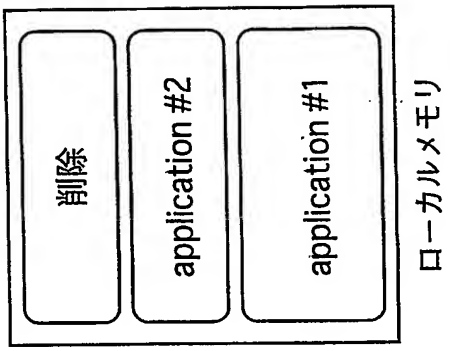


図59

(a) データ管理テーブル

生存区間	アプリケーションID	ロード属性	読込優先度	グループ属性
title #1	application #1		mandatory	—
title #1	application #2		optional	group #1
title #1	application #3		optional	group #1

(b)



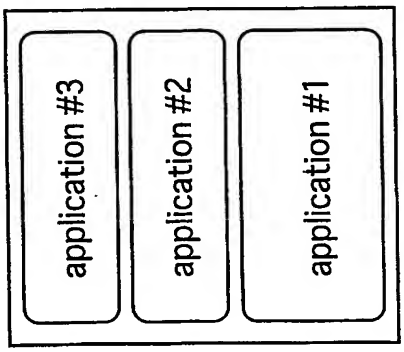
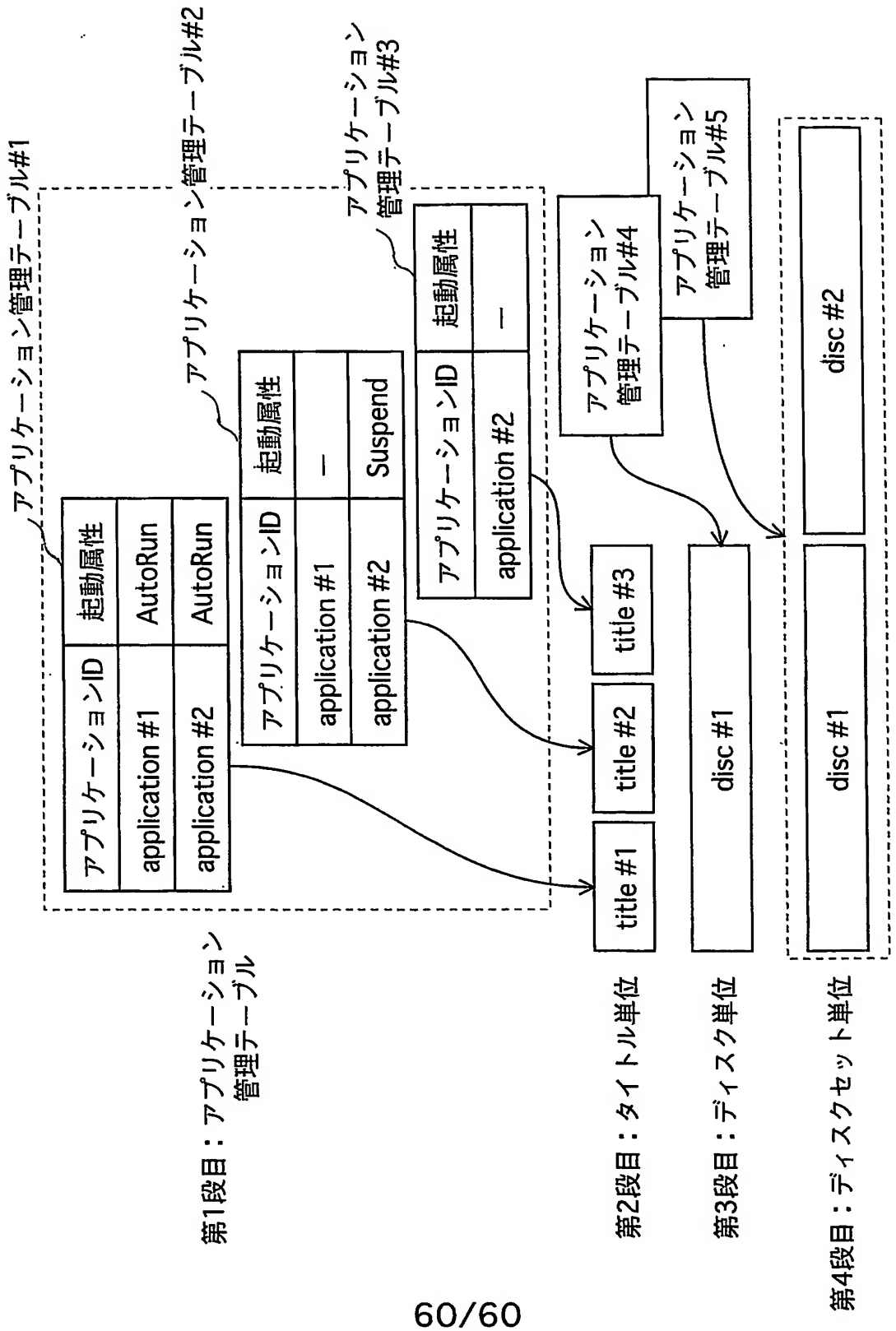


図60


アプリケーション管理テーブル#1
アプリケーション管理テーブル#2
アプリケーション管理テーブル#3
アプリケーション管理テーブル#4
アプリケーション管理テーブル#5
アプリケーションID
起動属性
application #1
application #2
AutoRun
AutoRun
—
Suspend
application #2
—
第1段目：アプリケーション管理テーブル
第2段目：タイトル単位
第3段目：ディスク単位
第4段目：ディスクセット単位
title #1
title #2
title #3
disc #1
disc #1
disc #2

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2004/015335

A. CLASSIFICATION OF SUBJECT MATTER
Int.Cl[7] G11B20/10, G11B20/12, G11B27/00, G11B27/10, G06F19/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
Int.Cl[7] G11B20/10, G11B20/12, G11B27/00, G11B27/10, G06F19/00

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Toroku Jitsuyo Shinan Koho	1994-2005
Kokai Jitsuyo Shinan Koho	1971-2005	Jitsuyo Shinan Toroku Koho	1996-2005

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	JP 2002-369154 A (Matsushita Electric Industrial Co., Ltd.), 20 December, 2002 (20.12.02), Full text; Figs. 1 to 39 & WO 02/082810 A1	1-7
A	WO 00/078043 A (WINK COMMUNICATIONS, INC.), 21 December, 2000 (21.12.00), Claim 3; Fig. 1 & WO 00/078043 A1	1-2
A	JP 2003-230104 A (Matsushita Electric Industrial Co., Ltd.), 15 August, 2003 (15.08.03), Par. No. [0004]; Fig. 1 & WO 03/047244 A1	5

☐ Further documents are listed in the continuation of Box C.

☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search 06 January, 2005 (06.01.05)	Date of mailing of the international search report 25 January, 2005 (25.01.05)
Name and mailing address of the ISA/ Japanese Patent Office Facsimile No.	Authorized officer Telephone No.

Form PCT/ISA/210 (second sheet) (January 2004)

国際調査報告

国際出願番号 PCT／JP2004／015335

A. 発明の属する分野の分類（国際特許分類（IPC））

Int. Cl' G11B 20／10 G11B 20／12 G11B 27／00 G11B 27／10
G06F 19／00

B. 調査を行った分野

調査を行った最小限資料（国際特許分類（IPC））

Int. Cl' G11B 20／10 G11B 20／12 G11B 27／00 G11B 27／10
G06F 19／00

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報 1922－1996年
日本国公開実用新案公報 1971－2005年
日本国登録実用新案公報 1994－2005年
日本国実用新案登録公報 1996－2005年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

C. 関連すると認められる文献

引用文献のカテゴリー＊	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求の範囲の番号
A	JP 2002-369154 A（松下電器産業株式会社） 2002. 12. 20 , 全文, 第1-39図 & WO 02/082810 A1	1-7
A	WO 00/078043 A（ウィンク・コミュニケーションズ・インコーポレーテッド）2000. 12. 21 , 【請求項３】, 第1図 & WO 00/078043 A1	1-2
A	JP 2003-230104 A（松下電器産業株式会社） 2003. 08. 15 , 【0004】, 第1図 & WO 03/047244 A1	5

☐ C欄の続きにも文献が列挙されている。

☐ パテントファミリーに関する別紙を参照。

＊ 引用文献のカテゴリー
「A」特に関連のある文献ではなく、一般的技術水準を示すもの
「E」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
「L」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）
「O」口頭による開示、使用、展示等に言及する文献
「P」国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献
「T」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの
「X」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの
「Y」特に関連のある文献であって、当該文献と他の１以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの
「&」同一パテントファミリー文献

国際調査を完了した日 06. 01. 2005

国際調査報告の発送日 25. 1. 2005

国際調査機関の名称及びあて先
日本国特許庁（ISA／JP）
郵便番号１００－８９１５
東京都千代田区霞が関三丁目４番３号

特許庁審査官（権限のある職員） 5Q 2946
前田 祐希

電話番号 ０３－３５８１－１１０１ 内線 ３５９０